알앤유 아두이노 프라임 키트 User Manual

ARDUINO PRIME KIT MANUAL

http://www.irnu.co.kr / www.irnumall.co.kr

아두이노 프라임 키트
사용자 매뉴얼
Version 1.0

http://www.irnu.co.kr
http://www.irnu.com.kr

Author	RNU CO,.LTD	Date First	2014 / 06 / 14
Name	KIT-A05	Start Ver	0.1
Update	2014 / 06 / 14	목록 정리, 추가 / Svn commit	
	2014 / 06 / 16~19	내용 추가 / Svn commit	
	2014 / 06 / 20	내용 보정, 추가, 오타 수정등등.. Svn commit	
	2014 / 06 / 21	내용 보강 및 추가. (적외선 센서 외)	
	2014 / 06 / 22~	Lcd1602 i2c 내용 보강. FND 내용 추가./보강 설명 이미지 보강, 릴레이 모듈 설명 보강	
	2014 / 06 / 23~	FND 설명 보강 예정.	
	2014 / 06 / 27	FND 내용 정리,	
	2014 / 06 / 29~~	4 Digit FND, 8x8 Dot Matrix 내용 보강 예정	
	2014 / 07 / 10	4 Digit FND, 8x8 Dot 다이렉트 와이어링 스크롤 예제 완료	
	2014 / 07 / 18~	스테퍼 모터 보강 구축 시작. 미 구축 예제 보강 시작 4-Digit 예제 설명 보강.	
	2014 / 07 / 20~	4 Digit FND 예제&설명 재 작성 완료. 사운드 센서 예제 보강 완료. 기타 항목 수정	
	2014 / 07 / 28~	LCD1602 다이렉트 연결 코드 & 설명 추가. 74HC595N 설명 보강&예제 추가. 스케치 설명 시작.	
	2014 / 08 / 10~	기본 예제 버튼/블링킹/풀업/풀다운, 초음파센서 예제 블루투스, L298N 초안 작성 중	
	2014 / 09 / 1~	LCD1602 예제 추가(시리얼 문자열 출력), 오타 수정, 문서 인덱스 추가.	
	2014 / 09 / 2	프라임 키트 설명 초안 RTC1302 모듈 설명 추가. 기타 항목 설명 사항 추가.	
	2014 / 09 / 26~	오타수정, 스테퍼모터 설명보강, 기타 수정	
	2014 / 10 / 01	5V 릴레이 설명 추가.	

	2014 / 10 / 15	RF 무선 통신, PWM 설명 수정.
	2014 / 10 / 22	LCD1602 I2C 적용 불가능 라이브러리 설명 제거. 오류문장 수정.
	2014 / 10 / 29	아날로그 포트에 대한 pinMode 설명 추가.
	2014 / 11 / 03	IR 수신 설명 보강.

목차

1 > 아두이노란?

아두이노는 “오픈 소스 기반의 하드웨어와 소프트웨어 개발 환경” 를 통칭합니다.
하드웨어, 소프트웨어 오픈 소스 기반에서 출발하여 오랜 시간 많은 분야에서 활용 되고 있습니다.
하드웨어는 아두이노 우노 / 메가 2560 / 나노 / 레오나르도 / 프로미니 등의 다양한 MCU 보드가 있습니다.
소프트웨어는 대표적으로 스케치 IDE 가 있습니다.

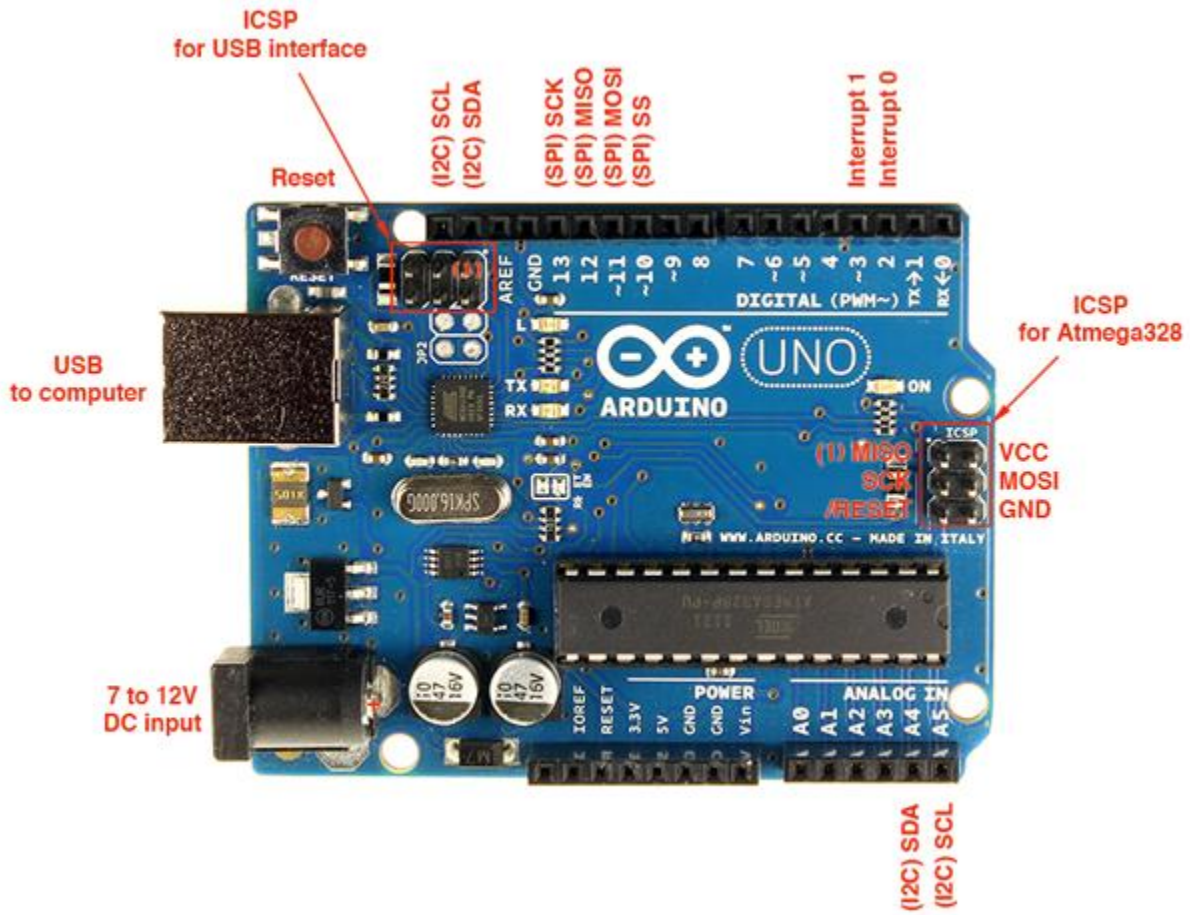
공식 아두이노 사이트는 **http://www.arduino.cc** 입니다.
아두이노와 스케치 IDE 를 사용하기 위해서는 자주 방문 하여야 할 사이트입니다.
아두이노를 활용한 기초부터 고급 기술까지 자세히 설명 되어 있습니다.
IDE 라는 용어의 의미는 통합개발환경을 말합니다.
Integrated Development Environment 입니다.
IT 선진국 대한민국을 비롯하여, 미국, 캐나다, 호주, 영국을 비롯한 유럽, 동남 아시아의 교육 선진국에서 아두이노를 기반으로 하는 하드웨어, 소프트웨어 기초 교육을 비롯한, 응용 분야, 예술분야, 취미까지 광범위하게 사용되고 있습니다.

2 > UNO R3 BOARD FOR ARDUINO X 1

아두이노 우노 R3 보드 입니다.
ATMEGA328P-PU 가 탑재된 형태의 마이크로 컨트롤러 보드입니다.
아트메가 328 칩을 사용한 제어 포트의 형태가 그대로 보드의 가장자리에 배치된 형태입니다.
보드 구성 이해를 하기 위한 핀맵 (Pin Map) 다이어그램 이미지입니다.

1.
2.

2.1 I2C & SPI & ICSP 핀맵 기능 요약 다이어그램.



ICSP for USB Interface: ATMEGA16U2 펌웨어 업로드 연결 ISP 포트.

아두이노 우노 Rev 3 보드는 ATMEGA16U2 사용됩니다.

ICSP for ATMEGA328P: ATMEGA328P 펌웨어 업로드 연결 ISP 포트.

SPI 포트: SCK, MISO, MOSI, SS
I2C: SDA, SCL 포트 A4, A5 포트. I2C 라는 단어와 IIC 는 같은 의미입니다. 상단의 참조 이미지의 왼쪽 위, 오른쪽 아래 2 곳 있습니다. Rev 3 에서 왼쪽 위 I2C 포트 추가 되었습니다.

AREF: ADC 처리시 기준 전압(V)입니다. 좀더 세밀한 또는 다른 기준의 ADC 값을 일괄적으로 변경하고자 할 때 사용할 수 있습니다.

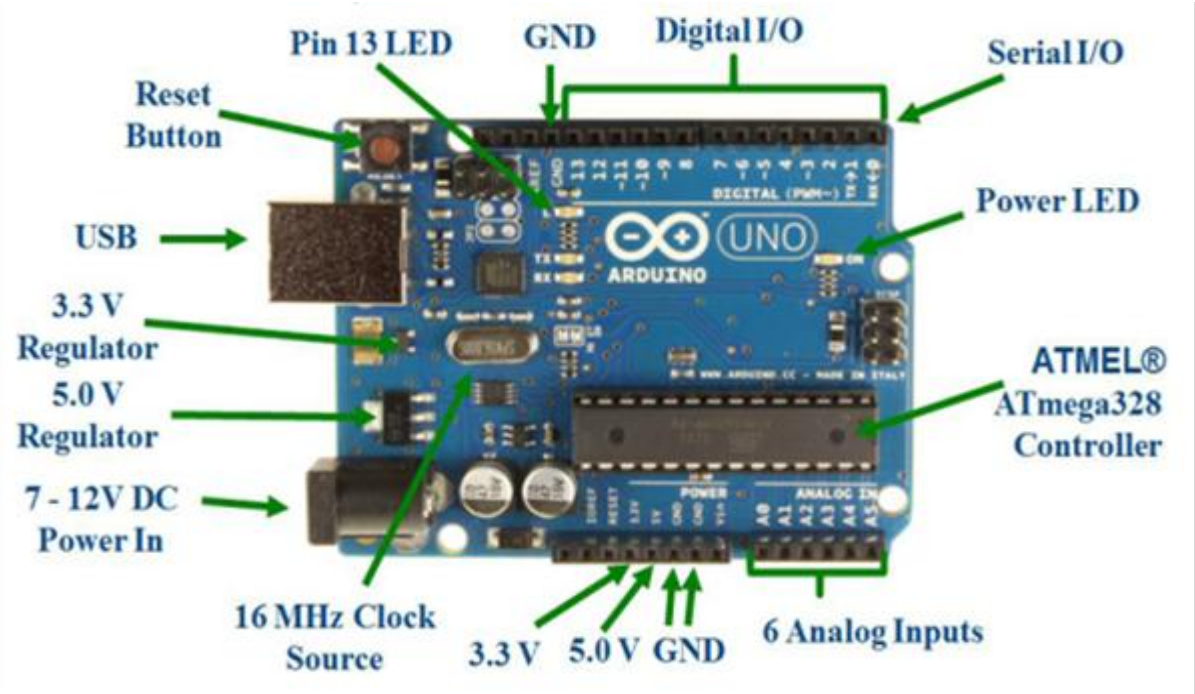
하드웨어 외부 인터럽트: INT0, INT1 2 개 사용 가능합니다.
D2, D3 에 연결하여 사용됩니다. 인터럽트 사용은 하드웨어 프로그래밍 요소 중 중요한 부분입니다.

7 to 12V DC Input: 7v to 12v DC 입력 전원 포트입니다. 보통 가정에서 많이 사용되는 9V, 12V DC 어댑터 그대로 사용하면 됩니다. DC JACK 규격은 5.5 파이, 내경 2.1 mm 입니다. **(Plug diameter 2.1mm ID, 5.5mm OD)**

보통 아두이노 우노 R3 의 추천 DC 어댑터는 9V 1A 어댑터 사용합니다.
물론, PC 와 USB 연결 후 동시에 사용하여도 무방합니다. 동시에 USB, DC 입력되는 경우에는 높은 입력 전압이 사용됩니다.

2.2 디지털 포트 & 아날로그 포트

아두이노 우노 R3 버전 아날로그 포트와 디지털 포트 설명 다이어그램입니다.



아날로그 Input 포트: A0 ~ A5, 6 개의 Analog Input Ports.

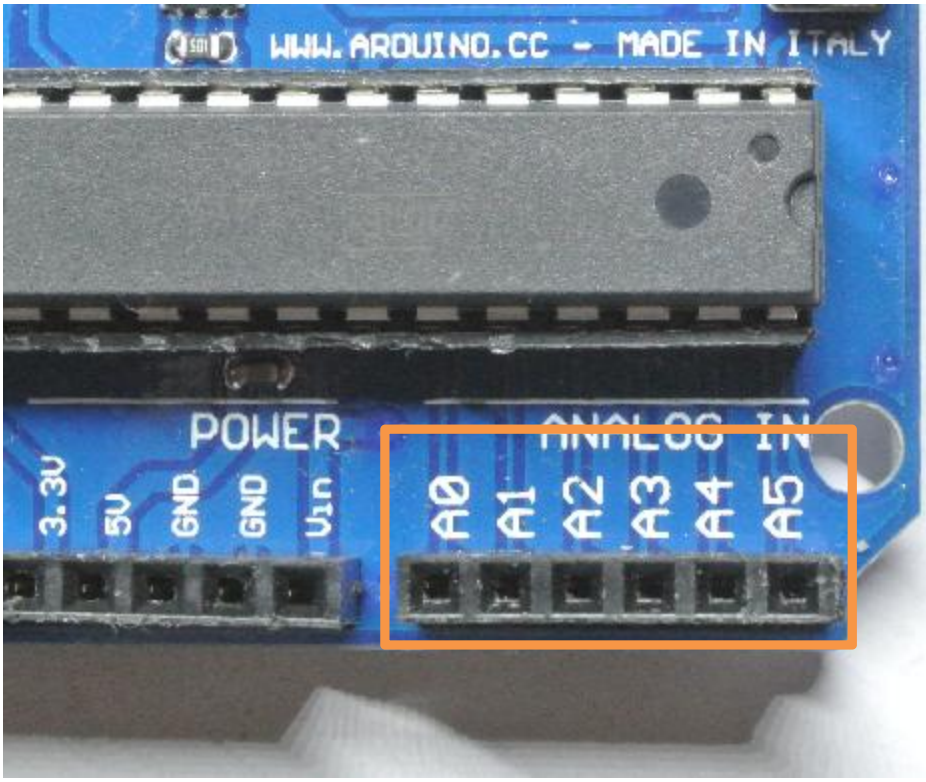
아날로그 입력 포트입니다. 각종 아날로그 센서들을 장착하여 측정값들을 가져올 수 있습니다. analogRead() 함수를 사용하여 아날로그 입력된 값을 받을 수 있습니다. 입력되는 측정값들의 범위는 0 ~ 1023 범위입니다.

디지털 In/Out 포트: 0~13, 14 Digital input/output Ports.

디지털 포트는 1, 0 의 값을 읽고 쓰기 할 수 있습니다. 0 은 LOW, 1 은 HIGH digitalRead(), digitalWrite() 함수를 사용하여 각각의 포트에 값을 읽기, 쓰기를 합니다.

2.3 아날로그 인풋 포트(ANALOG INPUT PORT)

6 개의 아날로그 입력 전용 포트입니다.
A0~A5 6 개의 입력 전용 포트입니다.
즉, 외부로부터의 아날로그 입력만 허용 됩니다. 보드에서의 출력 용도로는 사용 불가입니다..



스케치에서의 analogRead() 함수 사용시 입력 변수는 A0 ~ A5 사용됩니다.
1 개의 아날로그 인풋 핀에서는 1024 개의 아날로그 범위 값을 읽을 수 있습니다.
1024 개의 아날로그 범위 값을 읽을 수 있다는 것은 0 ~ 1023 범위의 integer type(정수) 입니다.
C/C++ 에서의 정수의 기본 인덱스는 0 부터 시작되므로 정수로 1024 개는 0~1023 이 됩니다.

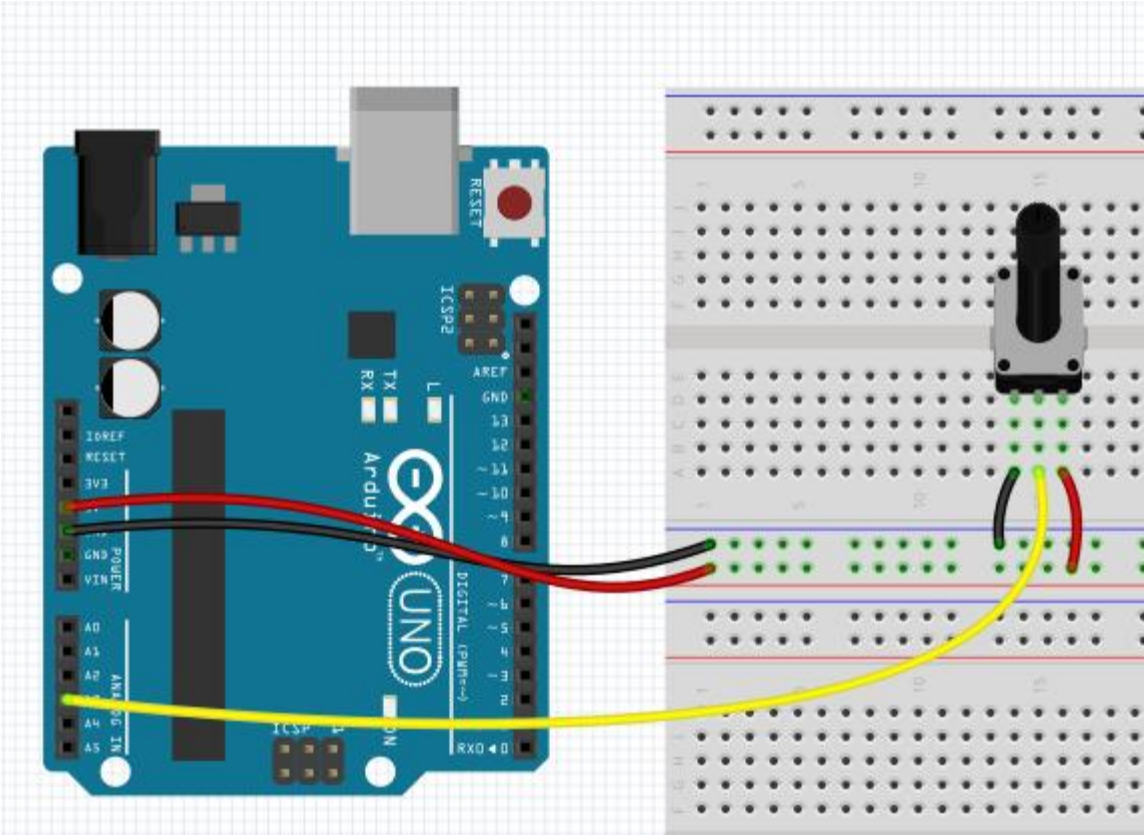
아두이노 공식 사이트에서도 상세히 설명 되어 있습니다.
http://arduino.cc/en/Reference/analogRead

예제 함수:

```
// 6 번째 아날로그 포트에서 값을 읽는다.
int a5_Value = analogRead(A5);

// 1 번째 아날로그 포트에서 값을 읽는다.
int a0_Value = analogRead(A0);
```

analogRead() 함수는 1024 범위의 값을 반환 합니다. (0 ~ 1023)

가변저항 10K 모듈을 사용하여 아날로그 입력 값을 알아봅니다.
가변저항을 돌려 입력되는 전압의 크기를 수치로 볼 수 있습니다.



예제 코드:

```
// 가변저항 10K 사용 예제 코드

int analogPin = 3;

int val = 0;

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

void loop()
{
      val = analogRead(analogPin);
      Serial.println(val);
}
```

가변저항의 값을 읽어서 시리얼 포트로 출력해 줍니다.

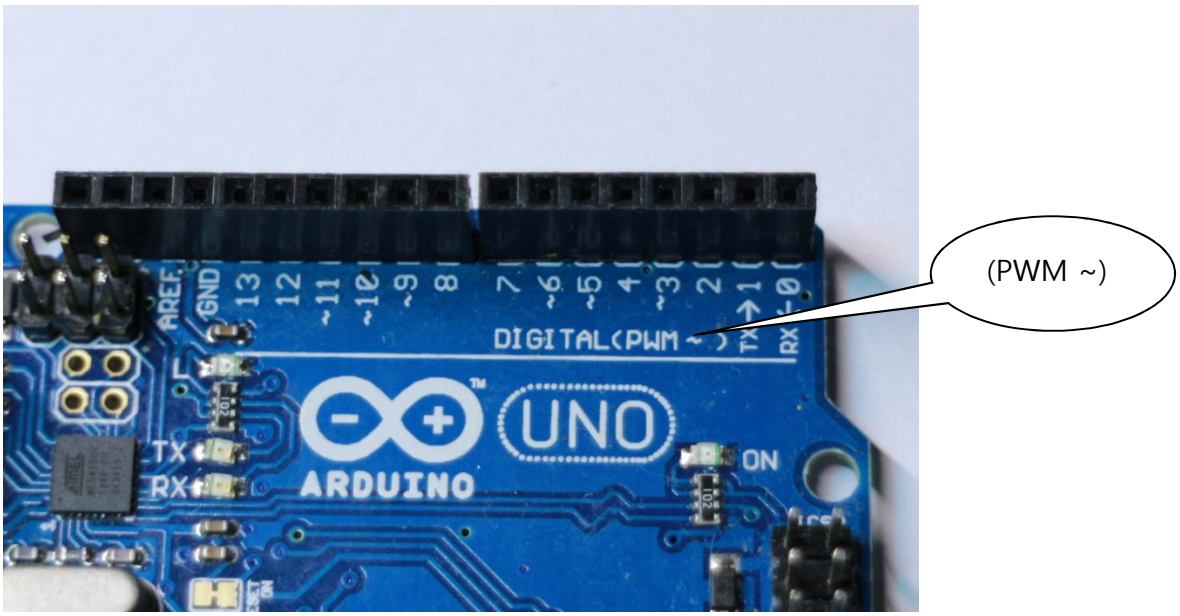
2.4 PWM 포트

우노 R3(ATMEGA328P-PU) 에서는 디지털 포트로 사용되는 14 개중 PWM 지원 포트가 있습니다. 아날로그 출력을 할 수 있습니다.

보드에 표시된 포트 명칭에 **(~)** 표시는 PWM 기능 표시입니다.

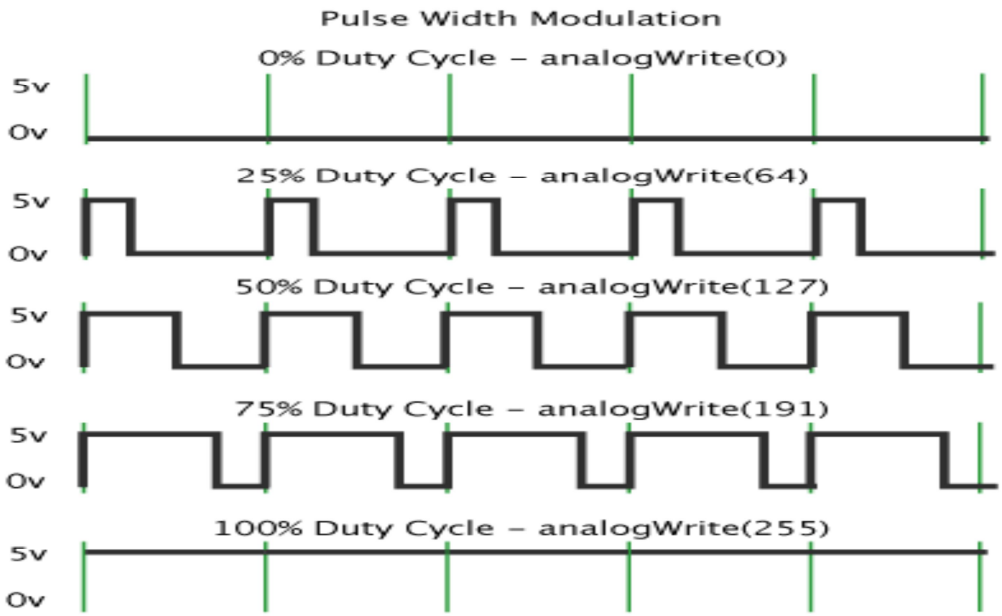
PWM - Pulse Width Modulation – 펄스폭 변조.

펄스폭 변조 발생시켜 디지털 출력으로 0 과 1 출력을 아날로그인 것처럼 출력할 수 있습니다.



~3, ~5, ~6, ~9, ~10, ~11 출력 핀으로 analogWrite() 함수를 사용 할 수 있습니다.

PWM 지원 포트(핀)에 256 범위의 값을 출력 할 수 있습니다.

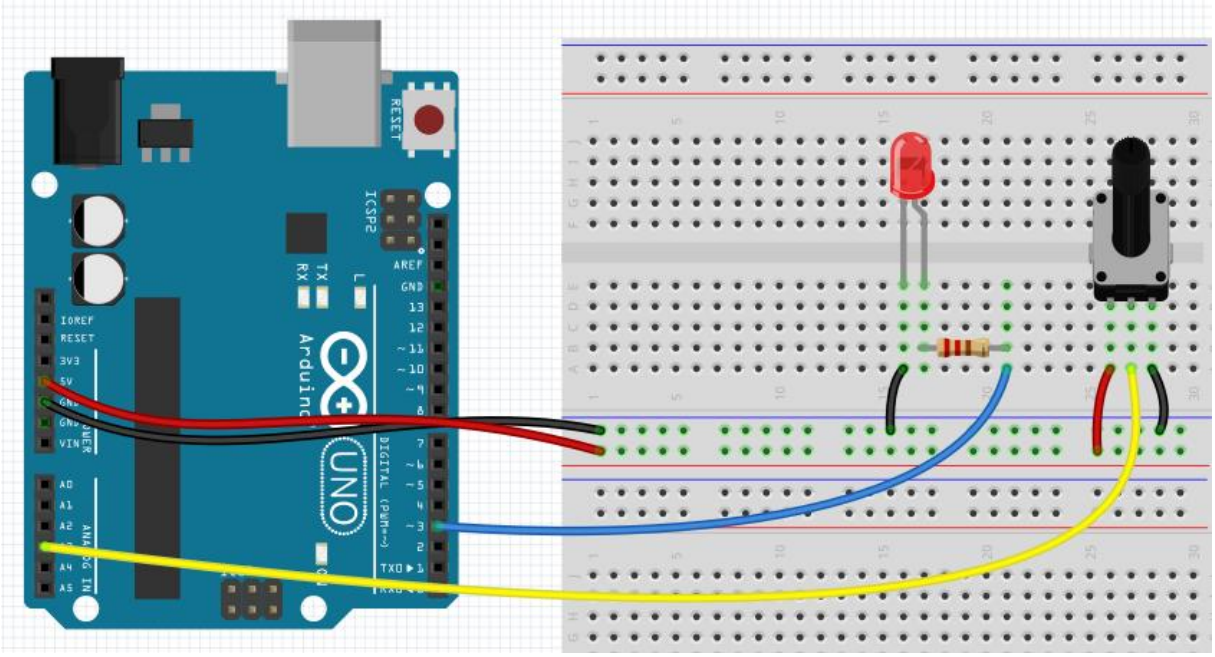


함수 정의:
analogWrite(pin,value)
pin : 포트 번호
value : duty cycle 값 : 0 ~ 255

analogWrite 함수는 256 개의 값을 사용 합니다. (0 ~ 255)
analogWrite 함수는 3,5,6,9,10,11 포트에만 적용됩니다.

물론 ~3, ~5, ~6, ~9, ~10, ~11 핀도 디지털 핀 방식으로 읽고 쓰기를 할 수 있습니다.
digitalWrite(), digitalRead() 함수 사용시 아날로그 범위 값은 무시되고 0, 1 값을 읽고/쓰기 됩니다.

가변저항 10K 와 LED (5 pi 크기) 예제
출처: http://arduino.cc/en/Tutorial/AnalogInput
가변저항 10K 와 LED 를 연결합니다.
LED 는 13 번 핀에 연결, 가변저항의 A0 (아날로그 입력 포트 1 번째)에 연결합니다.
가변저항을 돌려서 LED 의 밝기 조절할 수 있습니다.



예제코드)

```
int sensorPin = A3;     // 아날로그 포트 A3 에 가변저항 출력 연결
int ledPin = 3;         // LED 3 번 핀에 연결. PWM 가능 포트.

int sensorValue = 0;  // 가변저항에서 입력된 값 저장 변수.

void setup() {
    pinMode(ledPin, OUTPUT);    // 3 번 포트는 출력 전용으로
설정.
}

void loop() {
    // read the value from the sensor:
    sensorValue = analogRead(sensorPin);
```

```
    // 읽어들인 값을 pwm 범위 값으로 변환.
    int ledValue= map(sensorValue, 0, 1023, 0, 255);

    analogWrite(ledPin, ledValue);

    delay(100);              // 0.1 초 DELAY
}
```

2.5 디지털 14 핀 & 아날로그 포트 6 포트 사용 정의

디지털 포트와 아날로그 입력 포트는 위에서도 언급 설명 되어 있습니다.
우노 보드에서는 총 20 개의 디지털 입/출력 핀들이 제공되고 있습니다.
디지털 14 핀 D0 ~ D13
아날로그 6 핀 A0 ~ A5
만약, 프로젝트 진행시 디지털 입력/출력 포트가 부족할 경우에는 A0~A5 포트도 디지털 포트로 속성을 변경하여 사용 가능합니다.
pinMode 라는 함수를 사용합니다.

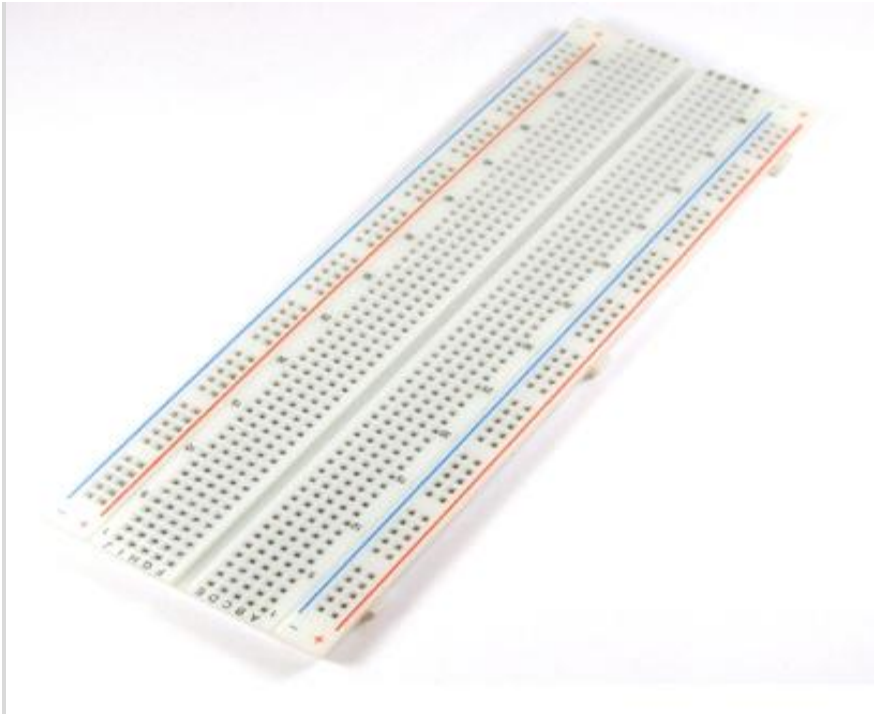
void pinMode(pin, mode)

pin 값으로 디지털 핀들은 0 ~ 13 값으로 할당됩니다.
아날로그 A0 ~ A5 핀들은 14 ~ 19 로 사용됩니다.
A0 와 A5 는 상수로 정의되어 있어 mode 값으로 같이 사용할 수 있습니다.
A0 혹은 14 를 사용하여도 같은 결과입니다.

```
void setup()
{
      //
      // 아날로그 포트 1st 포트를 디지털 i/o 포트로 사용.
      //

      // 아래 2 줄의 코드는 같은 의미입니다.
      pinMode(14,OUTPUT); //
      pinMode(A0,OUTPUT);
}

void loop()
{
      // 아래 2 줄의 코드는 같은 의미입니다.
      digitalWrite(14,HIGH);
      // 또는
      digitalWrite(A0,HIGH);
}
```

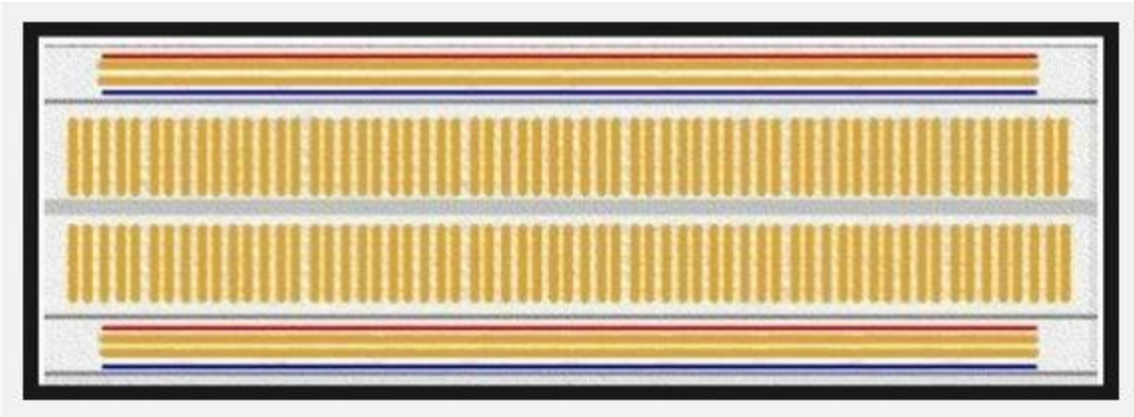
3 > HIGH QUALITY AND LARGE BREAD BOARD X 1

MB-102 브레드보드 입니다. 830 Points Solderless Breadboard
800 포인트 브레드보드입니다.



3.1 브레드 보드의 구조

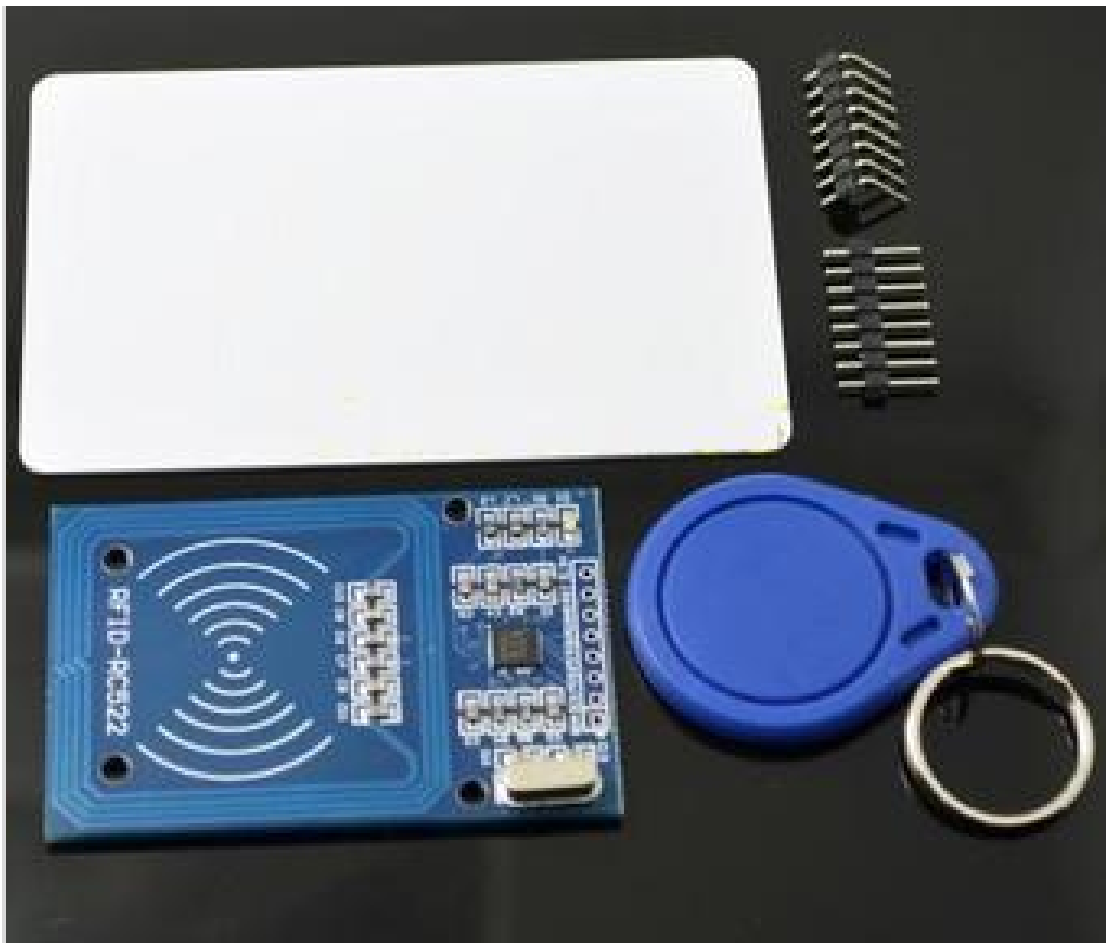
브레드보드는 전자 회로 적용 및 이해를 위한 용도로 많이 사용 됩니다.
브레드보드는 아래의 이미지와 같이 내부적으로 나란히 연결된 부분들이 있습니다.
빨간색과 파란색 표시선은 전원과 그라운드 표시입니다.



노란색 표시된 부분은 내부적으로 전기가 통하는 연결된 구조를 나타냅니다.
공통으로 연결된 부분에는 여러 개의 핀을 연결 가능하게 되어 회로를 구성할 수 있습니다. 즉, 연결된 부분에 점퍼 선을 연결하면 전기가 통하게 되어 동일한 연결 회로 구성이 가능합니다.

4 > RFID MODULE KIT

아두이노에서 사용 가능한 RFID 모듈키트입니다.
아두이노와 여러 MCU 보드에서 사용 가능한 RFID 키트입니다.



RFID (Radio-Frequency Identification) 는 전파를 이용한 근/원거리 통신 기술입니다.
RFID 기술은 산업/가전 하드웨어 제품에 널리 사용 되는 기능입니다.
RFID 제품 분야는 찾아보면 생각보다 많습니다. 대중교통 버스, 지하철, 아파트 주차장 입구, 출입 통제되는 입구, 공장 자동화 시스템 등에 많이 사용됩니다.
인식 태그 형태는 신용카드 크기가 주로 사용되며 열쇠 모양, 작은 크기의 마스터키, 부착형 스티커 등등 다양합니다.
보통 사용되는 RFID 태그와 판독기의 인식 거리는 5 Cm 정도입니다.
RFID 태그는 내부에 안테나와 작은 집적 회로가 있습니다.

4.1 > RFID MODULE X 1

RFID-RC522 모듈 본체 입니다.

4.2 > IC KEYCHAIN X 1

열쇠 고리 모양의 RFID 태그입니다.
내부에는 작은 RFID 판독기에서 인식 가능한 칩이 들어 있습니다.

4.3 > NON-CONTACT TYPE IC CARD X 1

RFID 에 사용 가능한 태그는 ISO/IEC 14443-1:2008 또는 ISO/IEC 14443 입니다.
신용카드 크기입니다. 내부에도 RFID 모듈에서 인식 가능한 작은 칩이 들어 있습니다.

아두이노 보드와의 와이어링은 아래와 같습니다.

SPI 인터페이스 입니다.

RC522 보드 8 핀 표시 순서대로 나열 합니다.

RFID_RC522 모듈	아두이노 우노 / 프로 미니 / 나노
SDA	D10 SS
SCK	D13 SCK
MOSI	D11 MOSI
MISO	D12 MISO
IRQ	NC
GND	GND
RST	NC or D9 – (예제 라이브러리 참조)
3.3V	3.3V

아두이노 데모 예제 & 샘플 코드

https://github.com/miguelbalboa/rfid

위 주소에서의 예제 코드에는 우노 & 메가 보드에서의 사용 설명이 포함 되어 있습니다.

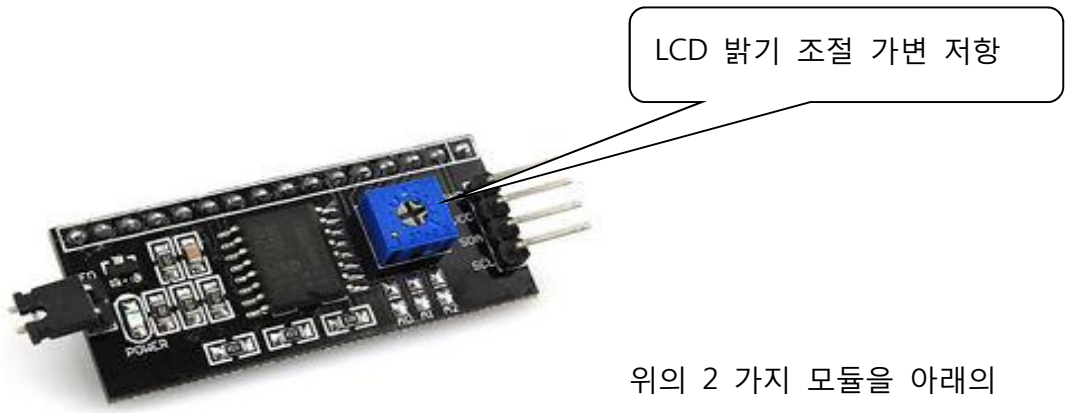
RFID-RC522 IC DataSheet:

http://www.igameplus.com/pds/gpshop/arduino/pdf/MFRC522.pdf

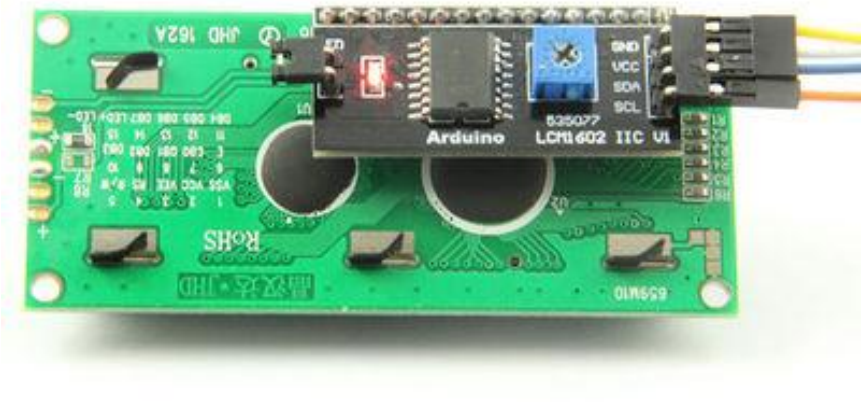
5 > LCD1602 I2C/IIC INTERFACE MODULE X 1

LCD 1602 모듈과 I2C 인터페이스 모듈입니다.
VCC 는 5V 입니다. 또는 3V3 전원 연결도 작동 되리라 봅니다.
LCD 1602 (16 x 2) 16 행 2 열의 문자 표시 모듈입니다.

LCD1602 16 핀 I2C 4 핀 인터페이스 모듈입니다.



위의 2 가지 모듈을 아래의 이미지처럼 연결해서 사용합니다.
(현재 키트에 포함 된 모듈은 미리 납땜으로 합체된 모듈입니다)



아두이노 우노 보드의 I2C 포트에 연결합니다.

I2C Address: 0x27

Pin Definition: VCC、GND、SDA、SCL

1602 i2c 모듈	아두이노 우노
VCC	5V
GND	GND
SDA	A4
SCL	A5

5.1 I2C LCD1602 예제

I2C LCD1602 DataSheet

http://www.igameplus.com/pds/gpshop/arduino/pdf/LCD1602-I2C.pdf

I2C LCD1602 라이브러리 & 예제 코드 첫 번째

http://www.igameplus.com/pds/gpshop/arduino/shield_module/LiquidCrystal_I2Cv1-1.rar

라이브러리를 다운로드 후 스케치 myLibrary 디렉터리에 적절히 복사 해 줍니다.

```
#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()
{
      lcd.init();                              // initialize the lcd
      // Print a message to the LCD.
      lcd.backlight();
      lcd.print("Hello, world!");
}
void loop()
{

}
```

5.2 시리얼로 받은 문자열을 LCD1602 I2C 모듈로 출력

아래의 예제 코드를 업로드 합니다.
코드의 내용은 시리얼로 입력 받은 문자열을 LCD1602 I2C 인터페이스 모듈로 출력하는 코드입니다.
예제 코드를 테스트하기 위해서는 업로드 후 스케치 IDE 의 시리얼 모니터창을 열고 임의의 문자/숫자 들을 입력해 봅니다.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27,16,2);   // set the LCD address to 0x27 for a 16 chars and 2
```

```
line display

void setup()
{
  lcd.init();                    // initialize the lcd
  lcd.backlight();
  Serial.begin(9600);
}

void loop()
{
  // when characters arrive over the serial port...
  if (Serial.available()) {
    // wait a bit for the entire message to arrive
    delay(100);
    // clear the screen
    lcd.clear();
    // read all the available characters
    while (Serial.available() > 0) {
      // display each character to the LCD
      lcd.write(Serial.read());
    }
  }
}
```

위의 코드 적용시 깨진 문자처럼 나옵니다.
lcd.write() 함수는 BYTE 값 출력이라 의도하지 않게 깨진 문자로 보입니다.
아래의 예제코드는 시리얼로 받은 BYTE 값을 프린트 가능하게 변환하는 부분 추가된 코드입니다.
아래의 예제 코드와 위의 예제 코드를 비교해 보시기 바랍니다.
시리얼로 입력된 전체 문자열들을 LCD1602 로 출력해주는 코드입니다.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27,16,2);   // set the LCD address to 0x27 for a 16 chars and 2
line display

void setup()
{
  lcd.init();                       // initialize the lcd
  lcd.backlight();
  Serial.begin(9600);
}

void loop()
{
  // when characters arrive over the serial port...
  if (Serial.available()) {
    // wait a bit for the entire message to arrive
    delay(100);

    // clear the screen
    lcd.clear();

    String szTemp;
    // read all the available characters
    while (Serial.available() > 0) {
     // display each character to the LCD
      char cRead=Serial.read();
      szTemp+=cRead;

    }
```

```
    if( szTemp )
    {
      lcd.print(szTemp);
    }
  }
}
```

5.3 LCD1602 다이렉트 와이어링

LCD1602 모듈을 위의 I2C 변환 모듈 보드를 사용하지 않고 LCD1602 본체에 바로 연결을 해서도 사용할 수 있습니다. 다만, 연결 핀 수가 많아 지는 단점은 있지만, 다른 용도로의 사용시 용이하게 적용 할 수 있습니다.

LCD1602	아두이노 우노
VSS	GND
VDD	5V
V0	가변 저항 10K
RS	12
RW	11
E	10
D0	NC
D1	NC
D2	NC
D3	NC
D4	5
D5	4
D6	3
D7	2
A	5V

K	GND

10K 가변 저항으로 밝기 조절을 합니다. 가변 저항은 우노 보드의 13 번 또는 원하는 핀에 연결 후 코드에 적용 합니다.
LCD 모듈의 V0 과 우노 보드의 13 번 핀에 가변 저항 연결의 이해는 가변 저항으로 LED 밝기 조절하는 개념과 동일합니다.

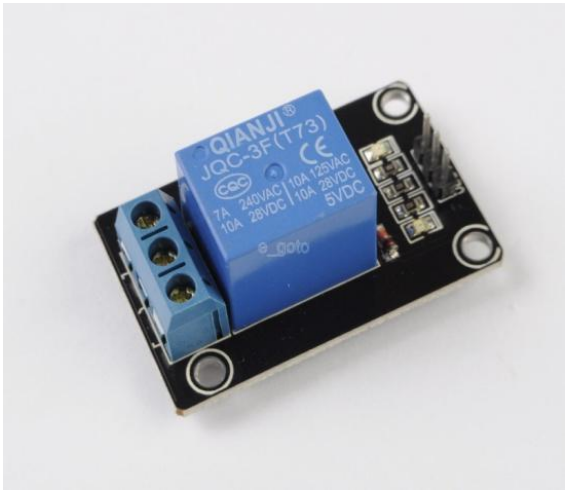
아래의 코드도 아두이노 라이브러리 LiquidCrystal 사용합니다.

```
#include <LiquidCrystal.h>

// Connections:
// RS (LCD pin 4) to Arduino pin 12
// RW (LCD pin 5) to Arduino pin 11
// Enable (LCD pin 6) to Arduino pin 10
// LCD pin 15 to Arduino pin 13
// LCD pins d0,d1,d2,d3 사용 안함.
// LCD pins d4, d5, d6, d7 to Arduino pins 5, 4, 3, 2
//
// 아두이노 13 번 핀에 10 K 가변 저항을 연결합니다.
const int BACK_LIGHT = 13; // Pin 13 will control the backlight

LiquidCrystal g_lcd(12, 11, 10, 5, 4, 3, 2);

void setup()
{
    pinMode(BACK_LIGHT, OUTPUT);
    digitalWrite(BACK_LIGHT, HIGH);       // Turn backlight on.
                  // Replace 'HIGH' with 'LOW' to turn it off.
    g_lcd.clear();                        // Start with a blank screen
    g_lcd.setCursor(0, 0);                // Set the cursor to the beginning
    g_lcd.print("Hello,");
    g_lcd.setCursor(0, 1);                // Set the cursor to the next row
    g_lcd.print("Have a nice day");
```

```
}

void loop()
{

}
```

6 > OPTO-COUPLE 1 CHANNEL 5V RELAY X 1

5V 1 채널 릴레이 모듈 입니다.



5V 신호로 AC 전원 연결 1 개를 온/오프 합니다.
릴레이 작동 시 약간의 틱, 틱 소음이 발생 되는 건 정상 입니다.

모듈 스펙은 아래와 같습니다.

Dimensions: 19mm x 15.5mm x 20mm
Coil Voltage: 5V DC
Operate Voltage: 3.75V DC
Release Voltage: 0.5V DC
Coil resistance: 70Ω (+/- 7Ω)
Rated Coil Power: 0.36W (48VDC 0.51W)
Insulation Resistance: ≥ 100MΩ
Dielectric Strength: Between coil and contacts ≥ 1500VAC 1min, between open contacts ≥ 750VAC 1min
Operate/Release Time: ≤ 10ms/5ms
Terminal Type: PCB mounting
Contact Form: 1H/1Z

Contact Resistence: ≤ 100mΩ (1A 6VDC)

Contact Material: AgCdO, $AgSnO_2$

Contact Capacity: 5A 240VAC/30VDC, 10A 240VAC/28VDC

NO - Normal Open 입니다. 평상시에는 비 접속(오픈) 상태입니다.

NC - Normal Close(ed) 입니다. 평상시에 접속된 상태입니다.

COM - AC 전원 케이블 2 개중 1 선을 연결하고 NO (or NC) 에 연결합니다.

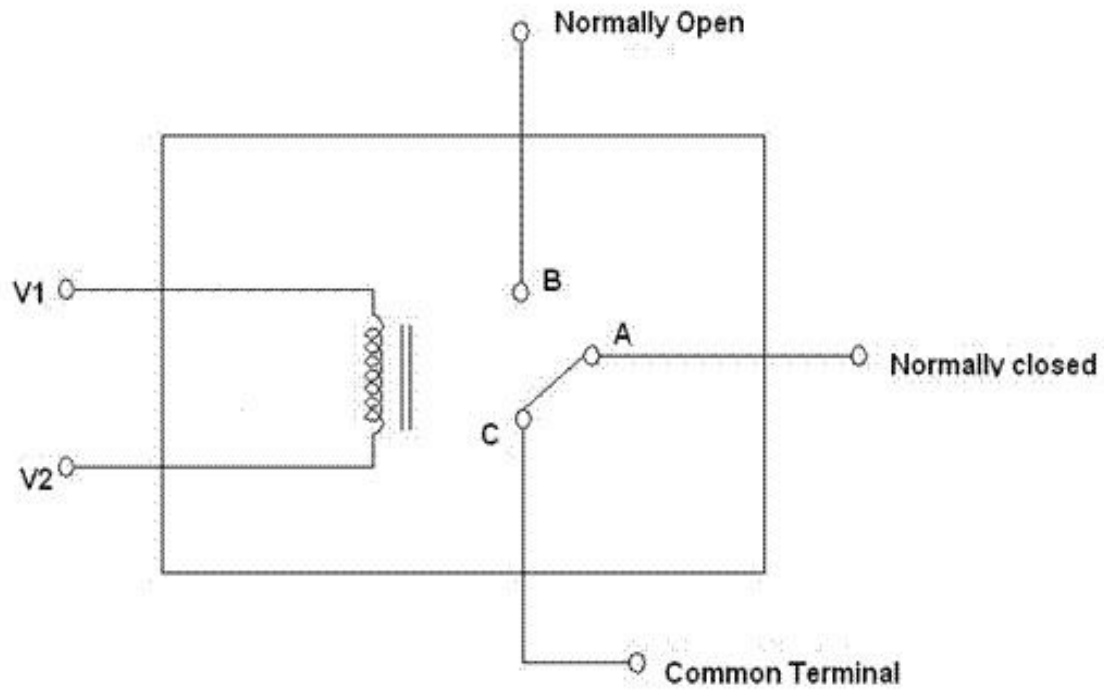
위 이미지에서 오른쪽 헤더 핀은 아두이노 보드에서 연결 됩니다.

VCC – 5V

GND – Ground

IN1 – 디지털 시그널 포트 연결

NO, NC 이해를 위한 회로도입니다.



5V-Relay 1 channel DataSheet:

http://www.igameplus.com/pds/gpshop/arduino/pdf/5v-relay-JQC-3F-T73.pdf

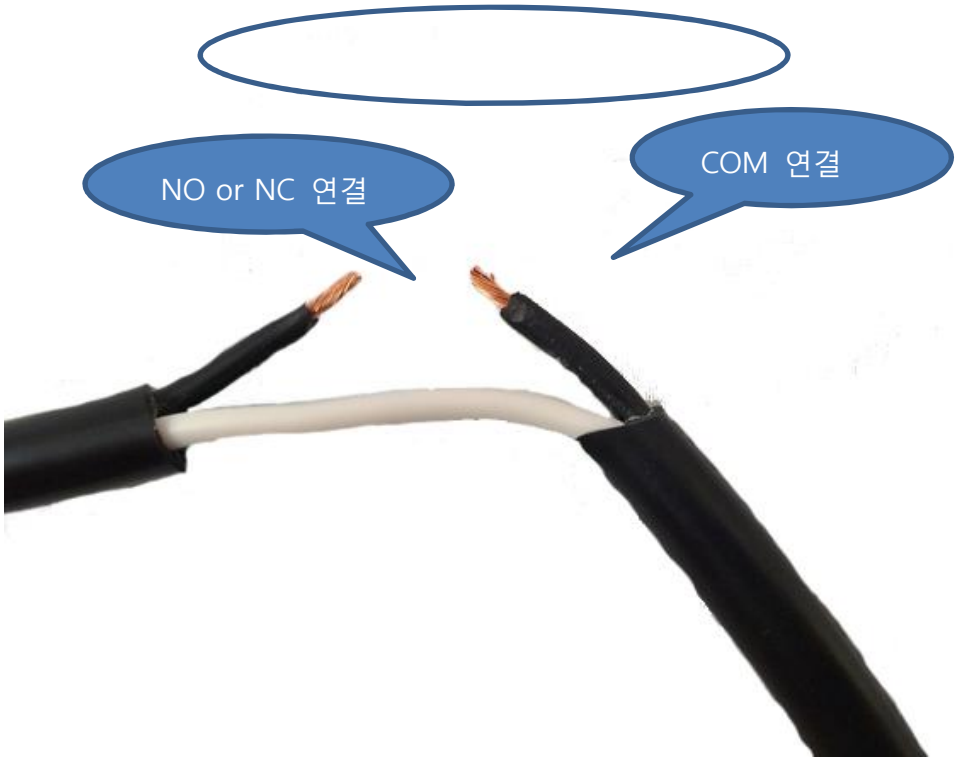
OR

http://www.igameplus.com/pds/gpshop/arduino/pdf/pcb-relay-T73.pdf

6.1 AC 전원선과 릴레이 모듈 연결 방법

먼저 AC 전원 연결 플러그를 빼서 안전하게 합니다.

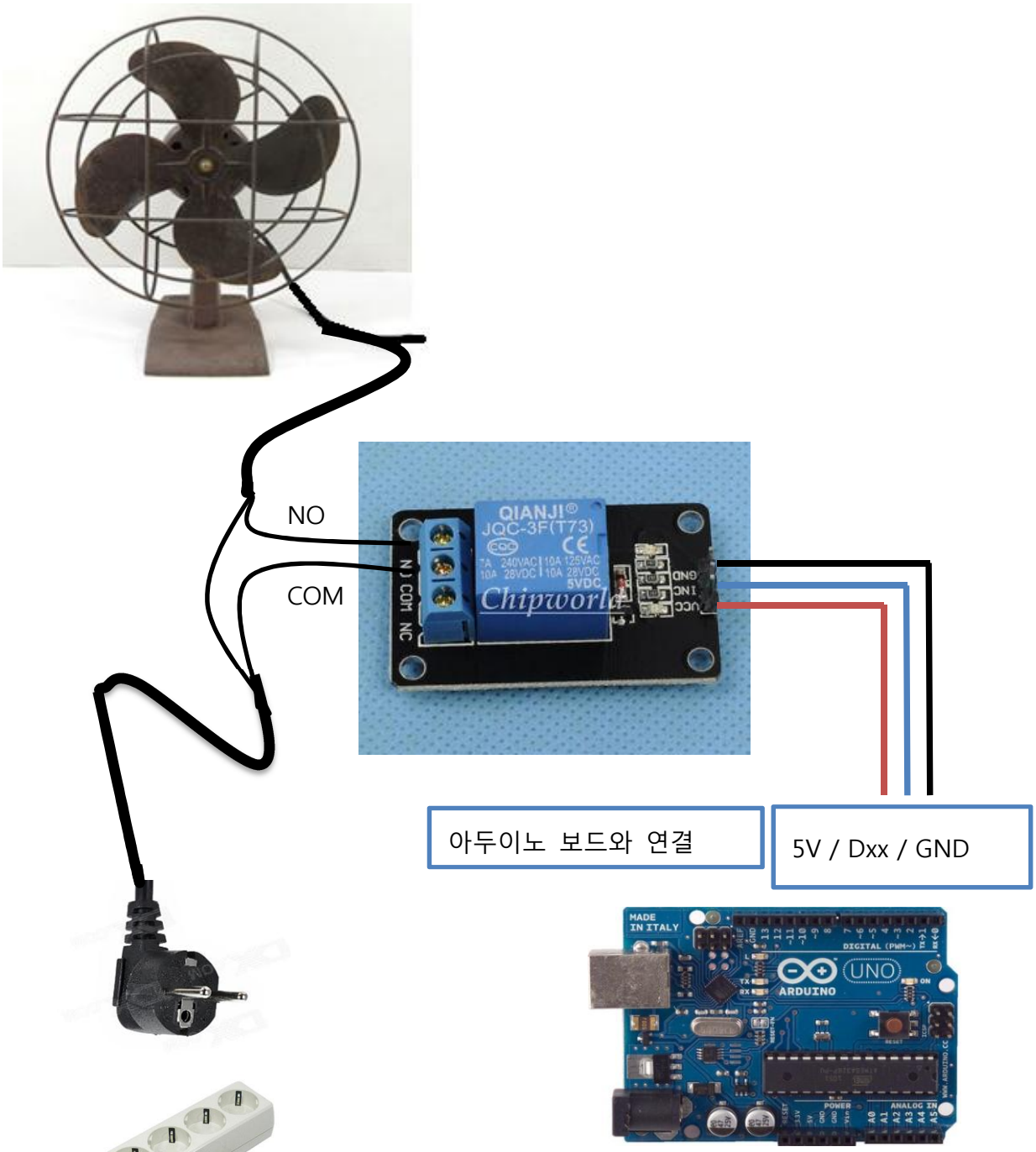
사용할 AC 전원 연결선 커트합니다. 아래의 이미지처럼 커트 합니다.



연결된 모습입니다. 테스트 용도일 경우 AC 220V 전원선을 연결하지 않고 낮은 전압의 LED 연결도 괜찮습니다.

선풍기 5V 릴레이 장착 아두이노 보드와 연결 참조 이미지

.



6.2 5V 릴레이 테스트 예제

5V 릴레이 테스트 하기 위해 스케치 프로그램의 기본 버튼 예제를 이용합니다.
버튼 예제는 10K 저항이 필요합니다. 아래의 이미지처럼 와이어링 합니다.
버튼을 누르면 LED ON, 안 눌린 버튼 상태이면 LED OFF 상태를 정확히 확인 및 테스트 합니다.

왼쪽 AC 전원 연결	오른쪽 헤더 핀	아두이노 보드
NO	GNC	GND
COM	IN1	디지털 포트
NC	VCC	5V 연결

.

릴레이 모듈의 5V & GND 아두이노 보드와 연결합니다.
릴레이 IN 포트와 아두이노 보드 13 번 포트와 연결합니다.
13 번 포트와 릴레이 IN 포트 연결하는 이유는 HIGH, LOW 신호를 보기 위한 용도입니다. 원하는 다른 디지털 포트 사용하여도 무방합니다.

릴레이 모듈 보드의 초록색(모듈에 따라 색상이 다른 색상일 경우도 있음)은 모듈 가동 상태입니다. 가동 중이면 ON 입니다.
릴레이 모듈 보드의 빨강색 LED 는 모듈에 따라 NC/NO 에 물려진 표시입니다. HIGH or LOW 일 경우 ON 됩니다.

>> 1 채널 5V 릴레이 사용을 이해하는 경우 2 채널, 4 채널, 8 채널 릴레이도 동일하게 사용됩니다. 모듈 자체에 제어포트와 전원연결부만 추가되는 형식입니다.

6.2.1 릴레이 테스트 코드 – 5 초마다 켜고, 끄기

5 초 반복으로 릴레이 켜고 끄기 테스트 코드입니다.
아두이노 디지털포트 13 번에 릴레이 모듈의 in1 or INC 포트와 연결합니다.
코드의 내용은 아두이노 보드의 디지털포트 13 번에 HIGH, LOW 신호 5 초마다 반복합니다.

예제코드:

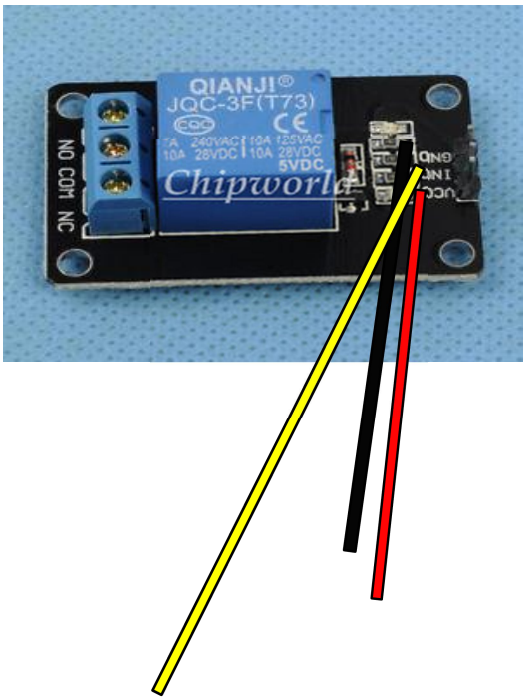
```
#define RELAY_ON 0
#define RELAY_OFF 1

#define Relay_1   13   // Digital Pin D 13

void setup()
{
      Serial.begin(9600); // 시리얼 포트로 메시지 보기 위해
      digitalWrite(Relay_1, RELAY_OFF);

      pinMode(Relay_1, OUTPUT);
      delay(5000); // 부팅시 5 초 지연.
}

void loop()
{
      // 릴레이 켜기 신호.
      digitalWrite(Relay_1, RELAY_ON);
      Serial.println("Relay On");
      delay(5000);                    // wait for 5 second

      // 릴레이 끄기 신호.
      digitalWrite(Relay_1, RELAY_OFF);
      Serial.println("Relay Off");
```

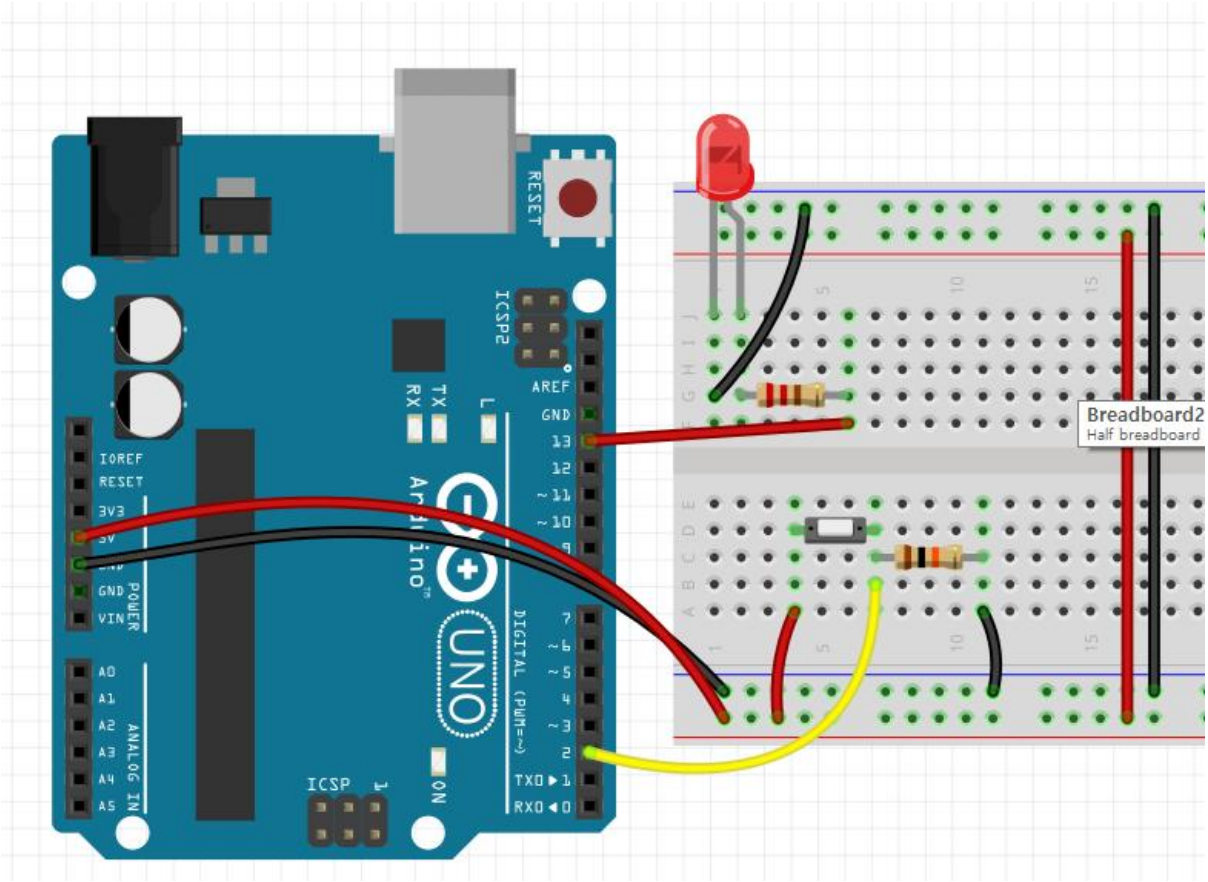
```
    delay(5000);
}
```

6.2.2 아두이노 버튼 누르면 등 켜지게 하기

버튼을 누르면 아두이노 보드 13 번 포트의 LED On 됩니다.
릴레이 모듈 보드의 초록색 LED 는 Off 됩니다.
버튼 포트는 PullDown 설정입니다.
(PullDown 버튼 회로 구성에 대한 설명은 차후에 기술합니다.)
AC 전원 220V 를 사용하는 만큼 정확한 사용법 숙지 후 안전하게 테스트 하시기 바랍니다.

5V 릴레이의 VCC → 우노 보드의 5V 와 연결.
5V 릴레이의 GND →우노 보드의 GND 와 연결.
5V 릴레이의 IN1 →우노 보드의 D13 와 연결.





위의 버튼 예제 브레드보드에 릴레이 연결된 브레드보드 회로 구성 이미지입니다.

버튼 예제 코드입니다.

```
// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = 2;     // the number of the pushbutton pin
const int ledPin =  13;      // the number of the LED pin

// variables will change:
int buttonState = 0;         // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
```

```
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}

void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
// if it is, the buttonState is HIGH:
//
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  }
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

먼저 위의 버튼 테스트 코드 및 회로도 구성 완성 후 5V 릴레이 모듈과 와이어링을 합니다.
5V 릴레이 모듈의 GND, VCC 와 아두이노 보드의 GND, 5V 연결 합니다.
아두이노 보드의 13 번 핀과 릴레이 모듈의 IN1 과 연결 합니다.

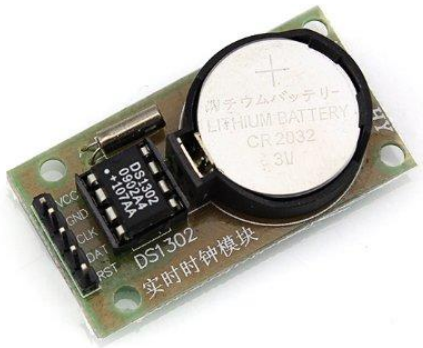
6.3 릴레이 활용

조도센서, 온도센서, 인체감지 센서 모듈과 연동하여 다양한 용도로의 사용 및 테스트가 가능합니다. A0~A5 에는 주로 아날로그 센서모듈들이 많이 사용됩니다.
인체감지 센서를 이용하여 자동으로 등 켜고/끄기 등은 쉽게 구현하실 수 있습니다.
조도센서를 이용하는 경우에는 어두울 경우 스위치를 켜고, 반대일 경우 스위치를 닫기 할 수 있습니다.
응용하기에 따라 많은 용도로 활용할 수 있습니다.

7 > DS1302 CLOCK MODULE X 1

RTC 모듈입니다.
시간 저장 및 저장된 시간 가져오기를 할 수 있습니다.
작동 전압 2V~5V 모듈 입니다.

1302 Arduino Uno
VCC -> 5V/3.3V
GND -> GND
CLK -> D4
DAT -> D3
RST -> D2



DS1302 모듈과 같이 배터리 내장 모듈들은 키트 포장시에 배터리 방지를 위해 배터리가 뒷면으로 되어 있는 경우도 있습니다. 사용시 다시 앞면으로 해주시기 바랍니다.

아두이노 예제 코드 및 설명
http://playground.arduino.cc/Main/DS1302 (모듈 형태만 다릅니다. 사용방법은 동일합니다.)

DS1302 DataSheet:

http://www.igameplus.com/pds/gpshop/arduino/pdf/DS1302_to_DS1302ZN.pdf

위의 아두이노 사이트의 예제가 복잡할 경우 아래의 주소에서 다운로드 받아서 사용하시면 됩니다.
http://www.igameplus.com/pds/gpshop/arduino/shield_module/RTC-DS1302-A-TYPE/DS1302.zip

DS1302.zip 을 다운로드 후 압축 해제 합니다. 스케치 IDE 의 myLibrary 디렉터리로 적절하게 복사 합니다. 또는 스케치 IDE 메뉴 -> 스케치 -> 라이브러리 가져오기 -> "라이브러리 추가" 선택합니다. 파일 선택 압축파일(ZIP), 파일 또는 폴더를 선택하면 자동으로 스케치 라이브러리에 추가 됩니다.
아래의 예제 코드를 적용 후 테스트 해 봅니다.

코드의 내용은 1 초마다 DS1302 모듈에서 가져온 시간을 시리얼 포트로 출력 합니다.

```
/* Define the DIO pins used for the RTC module */
#define SCK_PIN 4
#define IO_PIN 3
#define RST_PIN 2

/* Include the DS1302 library */
#include <DS1302.h>

/* Initialise the DS1302 library */
DS1302 rtc(RST_PIN, IO_PIN, SCK_PIN);

void setup()
{
  /* Clear the 1302's halt flag */
  rtc.halt(false);
  /* And disable write protection */
  rtc.writeProtect(false);
```

```
  /* Initialise the serial port */
  Serial.begin(9600);
}

/* Main program */
void loop()
{

  /* Set the time and date to 16:30 on the 3rd of September 2013 */
  rtc.setDOW(MONDAY);
  rtc.setTime(16,30,0);
  rtc.setDate(3, 9, 2013);

  /* Read the time and date once every second */
  while(1)
  {
    Serial.print("It is ");
    Serial.print(rtc.getDOWStr());
    Serial.print(" ");
    Serial.print(rtc.getDateStr());
    Serial.print(" ");
    Serial.print("and the time is: ");
    Serial.println(rtc.getTimeStr());

    /* Wait before reading again */
    delay (1000);
  }
}
```

8 > VOICE DETECTION MODULE X 1

LM393 Sound Detection Sensor Module

사운드 감지 센서 모듈입니다.

사용 되는 IC 칩은 LM393 입니다.

Chip: LM393, Electric condenser microphone; Operation voltage: DC 4-6V; 1-Way signal output, low level output

본 모듈의 연결 헤더 핀은 3 개입니다. (4 개짜리 모듈도 있습니다)

GND GND
OUT 아두이노의 아날로그 포트에 (A0) 연결합니다.
VCC 5V 작동 됩니다.(3v3 연결도 테스트 해 보시기 바랍니다)

사운드 센서 모듈에 가변 저항 조절 부분이 있습니다.
아래와 같은 코드를 작성&업로드 사운드 센서 아날로그 값을 출력 해봅니다.

스케치의 시리얼 모니터 창을 열어서 보면 연속해서 읽어 들인 값을 볼 수 있습니다.

```
int soundSensorPin=A0;              // for Analog input from sound breakout board.

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

void loop(){
   int sndValue = analogRead(soundSensorPin);
   Serial.println(sndValue); // 아날로그 입력 값 출력.
}
```

시리얼 모니터 창에서 가변 저항 조절부를 왼쪽/오른쪽으로 돌려보면 기본 측정값을 지정 합니다.
0 부터 1024 까지의 범위 입니다.
키트에 포함된 사운드 센서는 작은 소리에도 반응하도록 되어 있습니다.
기본 값을 38~40 정도 맞춰 봅니다.
사운드 센서의 목적 및 용도에 따라 회로 구성이 다양하게 나와 있으므로, 차후에 용도에 맞는 사운드 센서를 선택하는데 도움이 되리라 봅니다.

아래의 예제 코드는 사운드 발생시 LED 를 순차적으로 켜고, PWM 방식으로 부드럽게 OFF 하는 예제입니다.

```
//
// LED 는 3,5,6 번에 연결 후 테스트 해야 정확한
// 코드 이해가 됩니다.
```

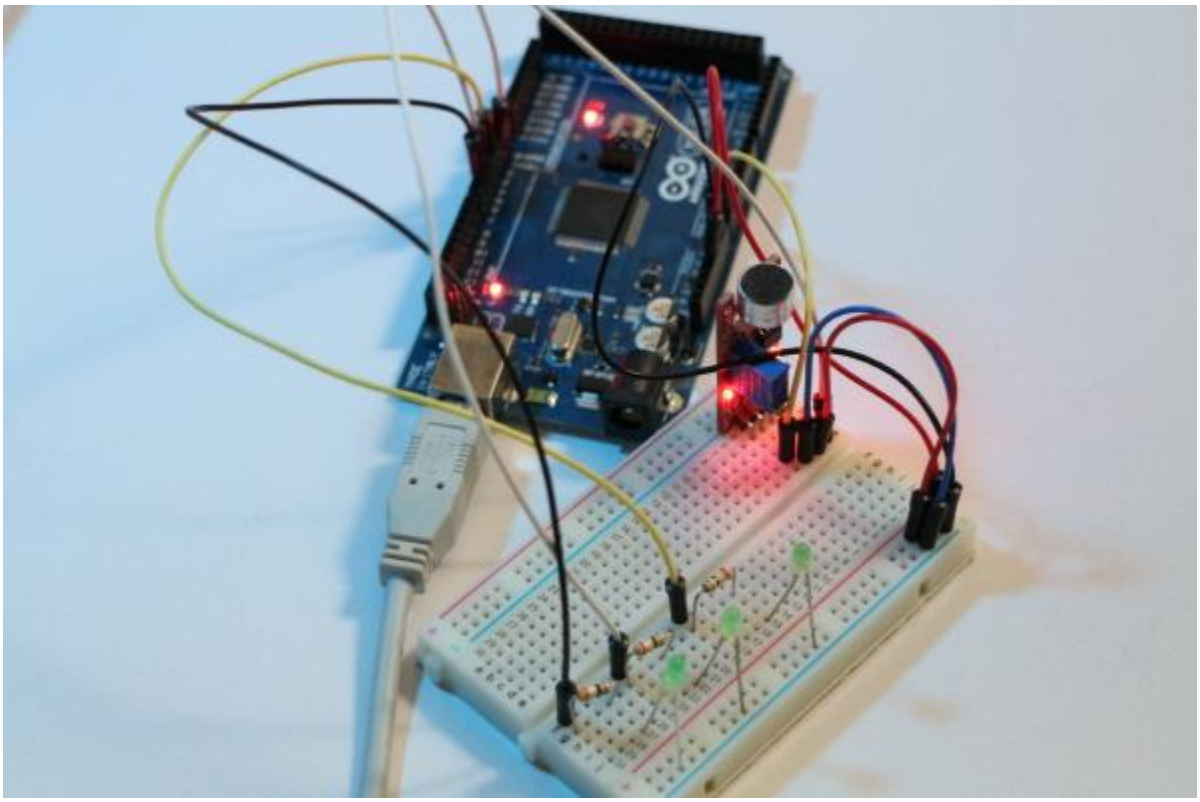
```
// 센서 모듈 <<--->> 아두이노 보드
// A0 <<-------->> A0
// GND <<-------->> GND
// VCC <<--------->> 5V
// D0  <<--------->> D2 // 4 핀 센서 모듈//
// 센서 모듈의 D0 핀 설명
// D0 -- HIGH
// D1 -- LOW (일정 강도의 소리가 전달될 때 LOW )
// 박수 소리 또는 단발성 소리가 나면 LED 가 순차적으로
// 점멸되는 예제입니다.
//
// sensing programming
// sound demo
//
// caution : base soundThreshold adjust 490 or you want.
// 사운드 센서 모듈의 가변저항을 왼쪽으로 계속 돌려서 490 정도로 맞추거나
// 센서 회로 구성에 따라 적절히 맞추어야 합니다.
//

//
int soundSensorPin=A0;              // for Analog input from sound breakout board.
//int soundSensorDigitalPin=2;     // for Digital input from sound breakout board.

int soundReading=0;
int soundThreshold=500;
int intensity[3]={0,0,0};
int LEDPins[3] = {3,5,6};
int numberOfPins=3;
int currentPin=0;
int fadeCounter=0;
int fadeDelay=50;

boolean switcher = true;
```

```
void setup()
{
      Serial.begin(9600);
      pinMode(soundSensorPin, INPUT);

      pinMode(soundSensorDigitalPin, INPUT);

      for(int i=0; i<numberOfPins;i++)
      {
            pinMode(LEDPins[i],OUTPUT);
      }
}

void loop(){
      // 4 핀 디지털 입력핀 사용시 코드.
      //    if(digitalRead(soundSensorDigitalPin)==LOW)
       // {
         //    Serial.println("LOW Digital input");
      //    }

       soundReading=analogRead(soundSensorPin);
       if(soundReading>soundThreshold)
       {
            Serial.println(soundReading);
            if(switcher)
            {
                  aboveThreshold(currentPin);
                  switcher=true;
            }
      }
      else
      {
            if(switcher)
            {
```

```
            belowThreshold();
            switcher=true;
            }
      }
}

void aboveThreshold(int cPin)
{
      switcher=false;
      if(intensity[cPin]<10)
      {
            intensity[cPin]=255;
            delay(50);
            currentPin=currentPin+1;
      }

      if(currentPin==numberOfPins)
      {
            currentPin=0;
      }
}

void belowThreshold()
{
      switcher=false;
      fadeCounter++;
      if(fadeCounter==fadeDelay)
      {
            fadeCounter=0;
            for(int i=0; i<numberOfPins;i++)
            {
                  analogWrite(LEDPins[i],intensity[i]);
            }
            for(int i=0; i<numberOfPins;i++)
```

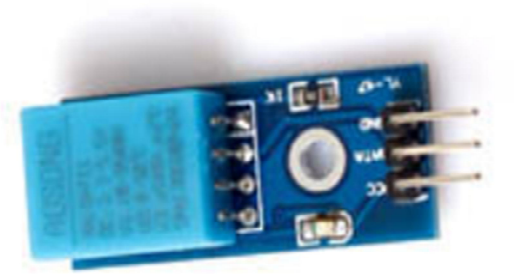
```
        {
            intensity[i]--;
            if(intensity[i]<0)
            {
                intensity[i]=0;
            }
        }
    }
}
```

아래는 위의 코드 구현된 예시 이미지입니다.



9 > TEMPERATURE AND HUMIDITY MODULE X 1

온습도 센서입니다.
온도, 습도 센서입니다. DHT11 센서를 사용하는 모듈 입니다.



작동 전원은 3.3~5.5V 모듈입니다..

DHT11 모듈	아두이노 우노
-	GND
Out	아날로그 포트
+	5V

DH11 DataSheet:
http://www.igameplus.com/pds/gpshop/arduino/pdf/Temperature_And_Humidity_Module_DHT11.pdf

아두이노 예제 및 라이브러리 참조
http://playground.arduino.cc/main/DHT11Lib

산업용/가전용 거의 모든 제품에 위와 비슷한 계열의 온도 센서가 사용 됩니다.
(의학/정밀 기기/소형 디지털 온도계의 칩셋들도 기본 온도 센서는 비슷하게 스키메틱 구성되어 있으나 오차 범위 측정(정밀도 오차 범위) 로직이 보강된 센서 모듈들은 고부가가치 센서 제품입니다)

10> LEVEL DETECTION MODULE X 1

물높이 측정 센서입니다.
(Raindrop Water Level / Height Depth Detection Sensor Module)

Product name: water level sensor
Operating voltage: DC 3.3V – 5.5V
Operating current: less than 20mA
Sensor type: analog
Detection area: 40 x 16mm
Production process: FR4 double-sided HASL
Operating temperature: 10°C - 30°C
Humidity: 10% - 90% non-condensing
Product dimensions: 62 x 20 x 8mm

위의 이미지의 빗금 부분을 물속에 넣고 물높이를 체크합니다.

아두이노 연결

Level 모듈	아두이노 우노
S	A0
+	5V
-	GND

예제코드: 아날로그 포트 0 번에서 값을 읽어 시리얼 프린트 합니다.

```
void setsup() {
    Serial.begin(9600);
}

void loop()
{
    // 아날로그 포트 0 번 읽기.
    Int sensorValue=analogRead(A0);
    Serial.println(sensorValue);
    //
    delay(200);
}
```

모듈 부분(커넥터 하부) 에만 물이 닿게 하고, 다른 부위(보드, 커넥터 포함)에는 물이 닿지 않게 합니다. 만약 물 높이 센서 측정 길이 이상으로 여러 개를 높이 단위에 따라(저수지 등등) 측정 하신다면 다른 여러 모듈 제품들도 있지만 방수 글리세린을 케이블 포함되게 충분히 덮어주는 식으로 가공하시면 충분히 WaterProof 제품을 구입하지 않고서도 사용 가능합니다.(다만 방수 글리세린은 물속에서 영구적으로 사용하지 못합니다)

10.1>> 여러 개의 아날로그 센서 사용시 주의사항.

A0 포트에는 아날로그 온도센서, A1 에는 사운드 센서, A2 에는 다른 아날로그 센서 등등 사용 할 경우에는 값을 제대로 읽어 오지 못하거나 이상한 값이 넘어오는 경우가 있습니다. 센서 자체의 Fetch 반환 기능이 늦거나, 아두이노의 ADC 처리 기능에도 최소한의 지연 시간이 발생됩니다.
위와 같은 문제 해결 방법은 여러 방법이 있겠지만, 간단하게 지연 함수를 사용 해 봅니다.
이런 경우는 아래의 예시 코드처럼 지연 함수를 사용해야 합니다.

```
int temp=analogRead(A0);   // 온도 센서
delay(100); // 적절한 값 테스트 요망
int snd_value = analogRead(A1);   // 사운드 센서
delay(100); // 적절한 값 테스트 요망
int temp=analogRead(A2);   // 물높이 센서
delay(100); // 적절한 값 테스트 요망
```

물론 위의 지연 함수(delay)를 사용하지 않고 제대로 값이 넘어온다면 delay 함수 사용 안 합니다.
여러 개의 아날로그 입력 핀 연결은 개별적으로 분리 되어 있으나 ADC(아날로그 디지털 변환기)라는 MCU 에서의 처리 부분이 여러 개의 아날로그 처리 시 제대로 처리되지 않을 수 있습니다.

10.2 ADC (Analog Digital Converter)

ADC (Analog Digital Converter)는 아날로그 신호를 디지털 신호로 변환해주는 역할 또는 장치입니다.
대부분 빛, 온도, 소리, 압력 등등의 변화 값들이 전압으로 바뀌면, 전압은 다시 기준 전압에 의해 일정 범위의 디지털 값으로 변경할 수 있습니다.
아날로그 입력->전압 변환->디지털 신호 출력 개념입니다.

MCU 의 ADC 기능에서는 위와 같은 과정을 여러 개의 포트에 대해 처리해 주어야 하므로 멀티플렉싱, 멀티플렉서 회로가 필요하게 됩니다. 결국, 여러 개의 아날로그 입력 처리시 약간의 지연 현상이 발생 될 수도 있습니다.

11> 4 * 4 KEYPAD MODULE X 1

4x4 버튼 키패드입니다. 16 개 버튼 입력 테스트 가능합니다.

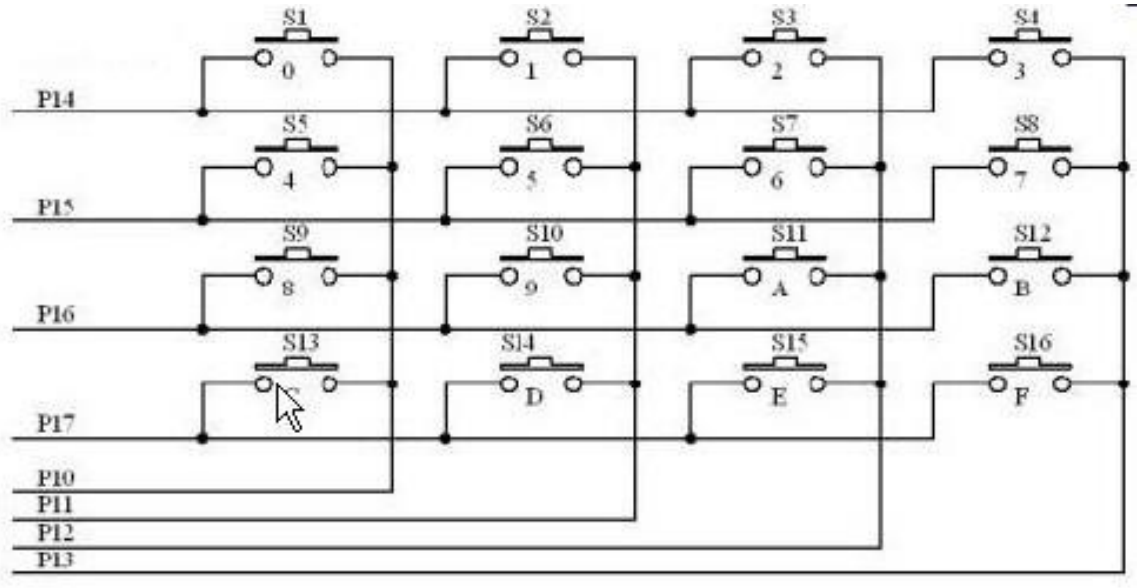
아래의 아두이노 예제 코드를 살펴보시기 바랍니다.
스케치 기본 라이브러리에는 KeyPad 모듈이 제공 됩니다.
아래의 예제 코드처럼 헤더 파일을 선언하여 사용합니다.

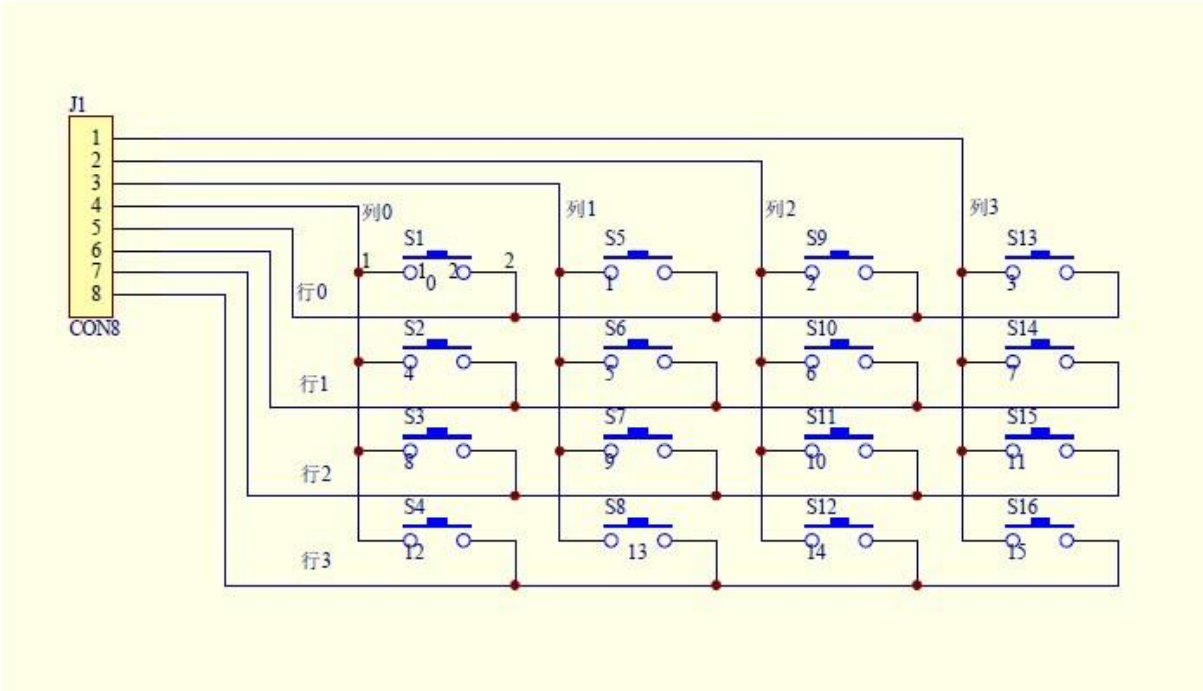
#include <Keypad.h> // keypad 라이브러리 사용

```
#include <Keypad.h>
```

```
const byte ROWS = 4;
const byte COLS = 4;
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
/* 아두이노 우노 보드 와이어링 핀입니다.*/
byte rowPins[ROWS] = {2,3,4,5};   //connect to row pinouts
byte colPins[COLS] = {6,7,8,9};      //connect to column pinouts

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

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

void loop()
{
      char key = keypad.getKey();

      if (key != NO_KEY)
      {
            Serial.println(key);
      }
}
```

스키메틱입니다.





아두이노 예제코드 & 참조

http://playground.arduino.cc/Main/KeypadTutorial

위의 제품은 차후에 전문적인 하드웨어 집적 회로도 설계시 기본적인 교육 회로도입니다. 이해를 충분히 하신다면 도움이 되는 회로도 입니다.(LED,FND 포함)

12> THREE-COLOR RGB MODULE X 1

RGB Color LED 모듈입니다.
RGB 값에 의한 LED 색상 변경 가능합니다. 5 파이 (5 Pi)크기의 RGB LED 가 적용된 모듈입니다.

RGB LED Module	Arduino Uno
GND	GND
R	11
G	10
B	9

아두이노 예제 코드입니다. 1 초마다 색상 변경해주는 코드입니다.
아두이노에 연결 시 9,10,11 주의하실 점은 PWM 포트에 연결합니다.

포트에 analogWrite() 함수를 사용하기 위해서는 PWM 지원 포트를 지정 해 주어야 합니다.

무대 조명, 빛 조명에서의 컬러 조절 및 배합은 상당히 예술적인 감각을 요하는 부분이기도 합니다. 위의 모듈로 기본적인 개념을 이해 하시고, 24 비트, RGB 의 색상 조절 개념을 숙지 하신다면 차후 전문가 or 취미 용도로도 활용 가능한 범위입니다.
아래의 예시 코드 적용 후 색상이 정확하지 않다면 와이어링 다시 체크 해 주세요.
코드 적용 후 1 초마다 색상 변경 됩니다.
빨간색/초록색/파란색/노란색/보라색 순서로 반복됩니다.

```
/*
Arduino - RGB LED
*/

int redPin = 11;
int greenPin = 10;
int bluePin = 9;

void setup()
{
	Serial.begin(9600);
	pinMode(redPin, OUTPUT);
	pinMode(greenPin, OUTPUT);
	pinMode(bluePin, OUTPUT);
}

void loop()
{
	Serial.println("red");
	setColor(255, 0, 0);   // red
	delay(1000);
	Serial.println("green");
	setColor(0, 255, 0);   // green
	delay(1000);
	Serial.println("blue");
	setColor(0, 0, 255);   // blue
	delay(1000);
	Serial.println("yellow");
	setColor(255, 255, 0);   // yellow
	delay(1000);
	Serial.println("purple");
```

```
        setColor(80, 0, 80);   // purple
        delay(1000);
        Serial.println("aqua");
        setColor(0, 255, 255);   // aqua
        delay(1000);
}
void setColor(int red, int green, int blue)
{
        analogWrite(redPin, red);
        analogWrite(greenPin, green);
        analogWrite(bluePin, blue);
}
```

13> XY JOYSTICK X 1

XY joystick 입니다.(Joystick Module Dual-axis XY for Arduino)
PS2 Game Joystick Module For Arduino

X 축, Y 축, 1 버튼 기능을 가진 조이스틱 모듈 입니다.
X, Y 축의 음직임을 2 개의 아날로그 신호로 받을 수 있습니다.
일반적인 조이스틱처럼 조종이 가능하며 조이스틱을 꾹 누르면(Z 축 기능) 버튼 기능이 됩니다.

와이어링 핀맵

XY Joystick Module	Arduino Uno
VCC	5V
GND	GND
VRx	아날로그 포트 A0
VRy	아날로그 포트 A1
SW	디지털 포트 Any

아두이노 사이트 튜토리얼 참조 (조이스틱 형태가 조금 다르게 보이지만 개념은 같습니다.)
http://www.arduino.cc/en/Tutorial/JoyStick

```
// # Description:
// # Modify the Sample code for the Joystick Module
// # Connection:
// #          X-Axis   -> Analog pin 0
// #          Y-Axis   -> Analog pin 1
// #          Z-Axis   -> Digital pin 3
// #

int JoyStick_X = 0; //x // A0
int JoyStick_Y = 1; //y // A1
int JoyStick_Z = 3; //key // D3

void setup()
{
  pinMode(JoyStick_Z, INPUT);
  Serial.begin(9600); // 9600 bps
}
void loop()
{
  int x,y,z;
  x=analogRead(JoyStick_X);
  y=analogRead(JoyStick_Y);
  z=digitalRead(JoyStick_Z);
  Serial.print(x ,DEC);
  Serial.print(",");
  Serial.print(y ,DEC);
  Serial.print(",");
```

```
    Serial.println(z ,DEC);
    delay(100);
}
```

14> SERVO X 1

SG90 모듈입니다.

서보 (서보 메커니즘) 모터입니다. 또는 Servo 단어 의미에는 Servant 라는 단어와 연관되어 주인 명령에 충실한 의미도 있다고 합니다.

위 모듈은 180 DEGREE (180 도 회전)입니다.

실제 사용시에는 180 도 적용 되긴 합니다. 데이터 쉬트와 실제 모듈과의 오류인지, 제조사 데이터 쉬트를 정확히 확인 하시기 바랍니다.

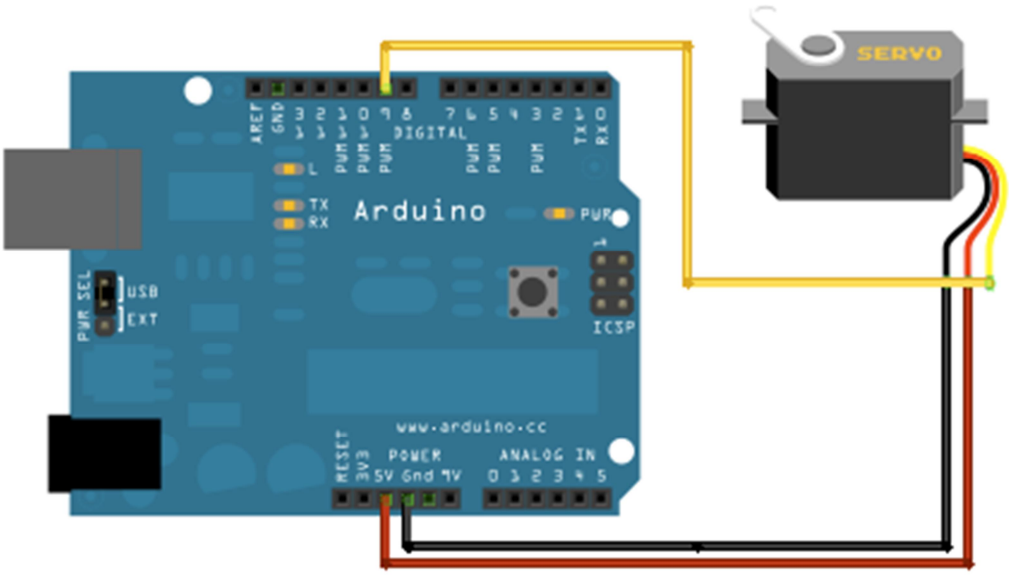
스케치 기본 라이브러리에서의 각도는 180 도로 구현되어 있습니다.

Specifications

Weight: 9 g
Dimension: 22.2 x 11.8 x 31 mm approx.
Stall torque: 1.8 kgf·cm
Operating speed: 0.1 s/60 degree
Operating voltage: 4.8 V (~5V)
Dead band width: 10 µs
Temperature range: 0 ºC – 55 ºC

모듈의 연결선은 3 개가 있습니다.

서보 모터 모듈	Arduino Uno
Black wire	GND
Red wire	5V
Blue or Yellow wire	9 or other PWM port

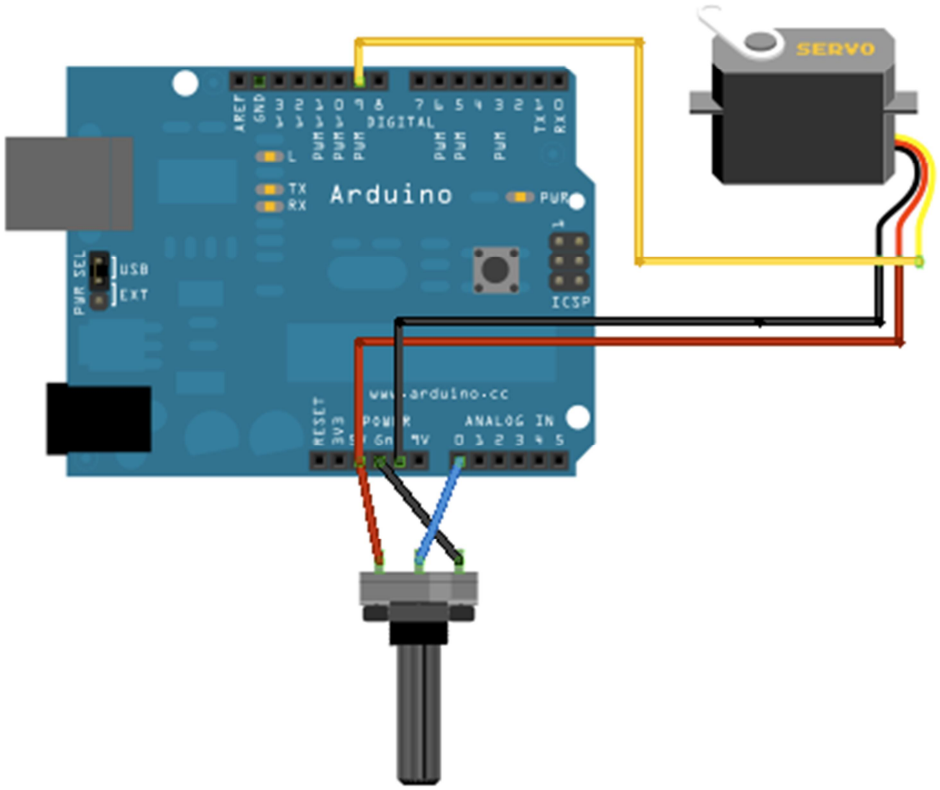


아두이노 사이트에 서보 모터 예제 및 설명이 있습니다.
1 번째 예제는 http://arduino.cc/en/Tutorial/Sweep
예제는 0 ~ 180 도 회전 후 다시 0 위치로 복구 되는 소스입니다.

```
// Sweep
// by BARRAGAN <http://barraganstudio.com>
// This example code is in the public domain.


#include <Servo.h>

Servo myservo;  // create servo object to control a servo
                // a maximum of eight servo objects can be created
```

```
int pos = 0;      // variable to store the servo position

void setup()
{
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}

void loop()
{
  for(pos = 0; pos < 180; pos += 1)  // goes from 0 degrees to 180 degrees
  {                                  // in steps of 1 degree
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the
position
  }
  for(pos = 180; pos>=1; pos-=1)     // goes from 180 degrees to 0 degrees
  {
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the
position
  }
}

// 2 번째 예제는 http://arduino.cc/en/Tutorial/Knob
// 포텐쇼미터(가변저항)의 아날로그 값을 받아 모터 컨트롤 하는 예제입니다.
// Controlling a servo position using a potentiometer (variable resistor)
// by Michal Rinott <http://people.interaction-ivrea.it/m.rinott>

#include <Servo.h>

Servo myservo;  // create servo object to control a servo

int potpin = 0;  // analog pin used to connect the potentiometer
```

```
int val;     // variable to read the value from the analog pin

void setup()
{
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}

void loop()
{
  // reads the value of the potentiometer (value between 0 and 1023)
  val = analogRead(potpin);
  // scale it to use it with the servo (value between 0 and 180)
  // 아날로그 입력된 0~1023 의 정수를 0~179 범위로 변경.
  val = map(val, 0, 1023, 0, 179);
  // sets the servo position according to the scaled value
  myservo.write(val);
  // waits for the servo to get there
  delay(15);
}
```



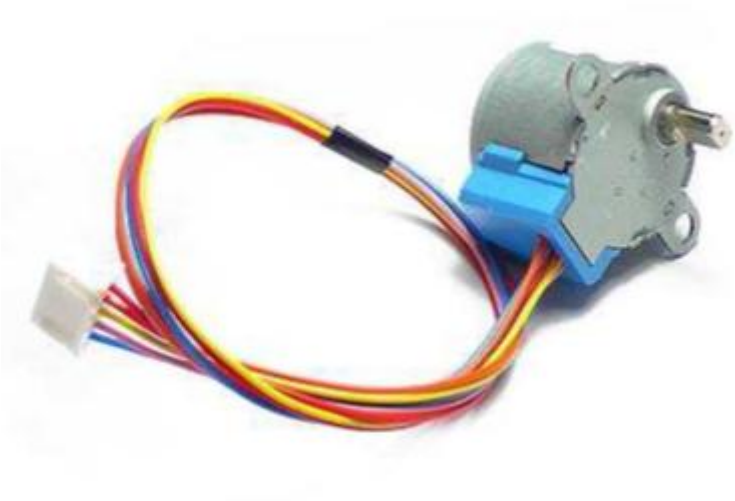
위의 예제코드에서 사용된 map 이라는 함수는 지정된 범위내의 입력된 값을 지정된 범위의 비율로 변환하여 반환하는 함수입니다.

스케치에서 사용되는 map 함수 본체입니다.

```
long map(long x, long in_min, long in_max, long out_min, long out_max)
{
  return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
}
```

15> STEPPER MOTOR & DRIVER BOARD X 1

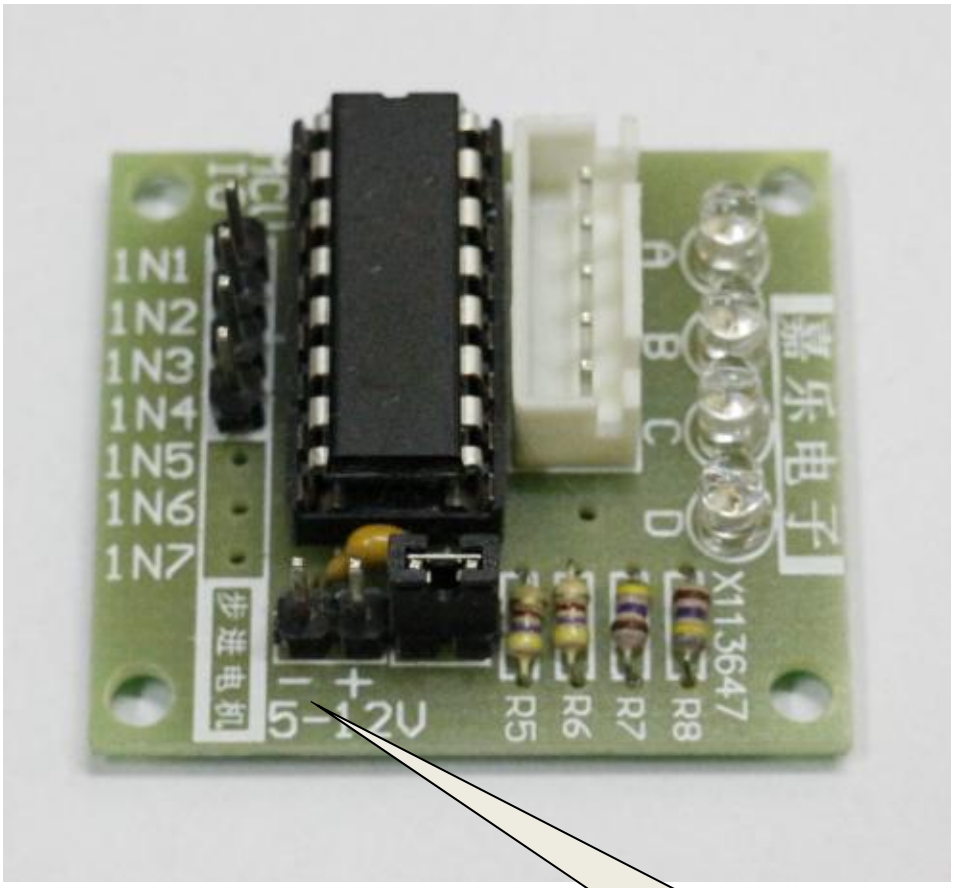
스테퍼 모터입니다. 스테퍼 모터는 펄스 입력으로 컨트롤합니다. 스테퍼 모터 또 다른 용어로는 펄스 모터라고도 합니다.
스테퍼 모터의 용도는 모터의 회전수와 회전량 등을 정밀하게 제어할 수 있습니다.
28BYJ-48 모델 스테퍼 모터입니다. ULN2003 스테퍼 모터 컨트롤 보드를 사용합니다.
ULN2003 컨트롤 IC 보드, 28BYJ-48 스테퍼 모터는 4 상(Phase 4) 모터입니다.
가동 전원은 5V 사용합니다.



스테퍼 모터의 사양은 아래와 같습니다.

Rated voltage ： 5VDC
Number of Phase 4
Speed Variation Ratio 1/64
Stride Angle 5.625° (360/64)
Frequency 100Hz
DC resistance 50Ω±7%(25℃)
Idle In-traction Frequency > 600Hz
Idle Out-traction Frequency > 1000Hz
In-traction Torque > 34.3mN.m (120Hz)
Self-positioning Torque >34.3mN.m
Friction torque 600-1200 gf.cm
Pull in torque 300 gf.cm
Insulated resistance > 10MΩ (500V)
Insulated electricity power 600VAC/1mA/1s
Insulation grade A
Rise in Temperature < 40K (120Hz)
Noise < 35dB (120Hz, No load, 10cm)
Model 28BYJ-48 – 5V

스테퍼 모터를 제어하는 ULN2003 제어 보드입니다.
핀 위치 참조 이미지입니다.



(-) GND
(+) 5V
우노와 연결합니다.

아두이노 연결:

ULN2003 모듈	아두이노 우노
IN1	8
IN2	9
IN3	10
IN4	11
+	5V 연결
-	GND

스테퍼 모터 사양은 보통 360 도(또는 제한된 각도) 회전에 대한 몇 회 분할을 할 수 있는 점이 중요합니다

28BYJ-48 스테퍼모터의 사양으로 스피드와 스텝을 계산해 봅니다.

Speed Variation Ratio (속도 변화 비율) : 1/64

Stride Angle 5.625°

Steps=Number of steps in One Revolution * Gear ratio
Steps=1 회전 단계 수 * 속도 변화 비율
Steps= (360°/5.625°) x 64 = 64 * 64 = 4096 스텝입니다.

15.1 스케치 예제 코드 1

다이렉트 제어 코드입니다.

```
/*
     BYJ48 Stepper motor code
     Connect :
     IN1 -> D8
     IN2 -> D9
     IN3 -> D10
     IN4 -> D11
     VCC . 5V Prefer to use external 5V Source
     Gnd
*/

 #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(115200);
    pinMode(IN1, OUTPUT);
    pinMode(IN2, OUTPUT);
    pinMode(IN3, OUTPUT);
    pinMode(IN4, OUTPUT);
    // delay(1000);
}

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--;
      }
  } // while

  Serial.println(time);
  Serial.println("Wait...!");
  delay(2000);
  Direction=!Direction; // 반전.
  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; }
```

```
}
```

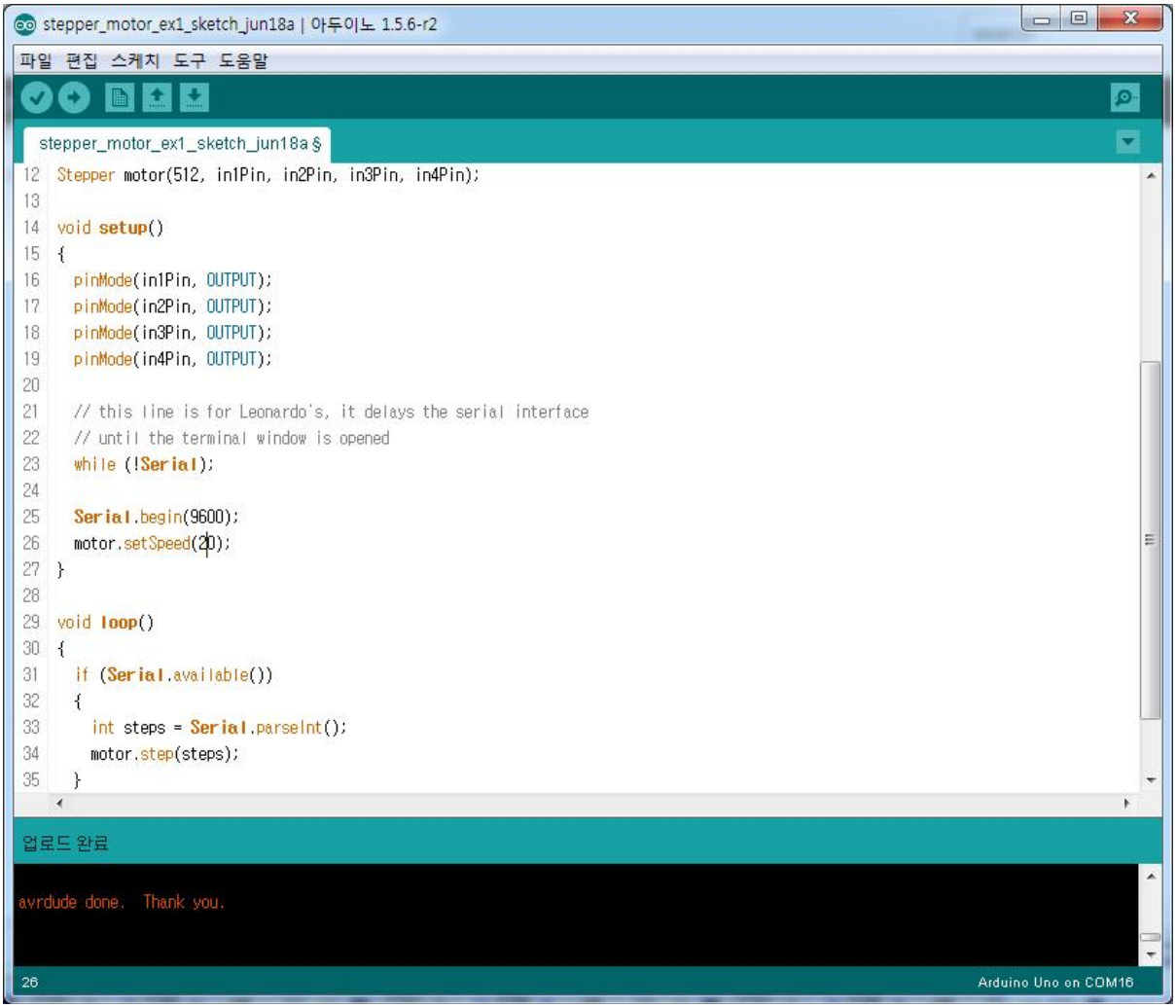
15.2 스케치 예제 코드 2

스케치 라이브러리 Stepper Motor Library 제어 코드입니다

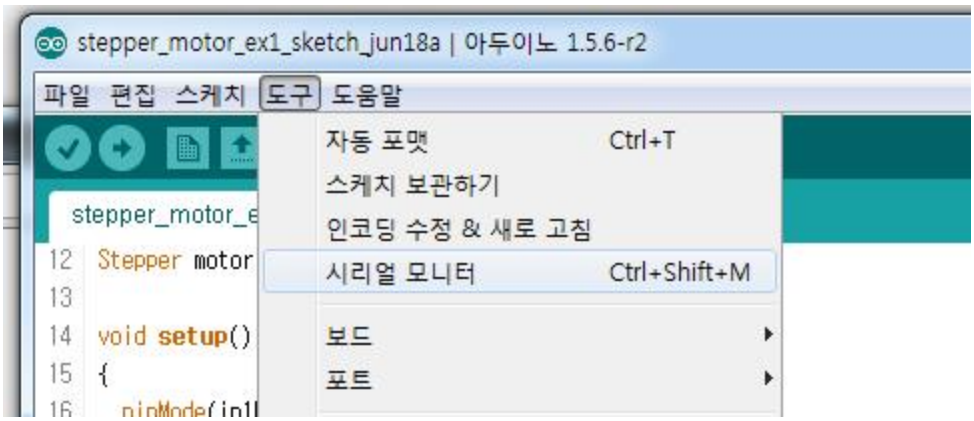
```
/*
스케치 라이브러리 Stepper 를 사용합니다.
*/
#include <Stepper.h>

int in1Pin = 8;
int in2Pin = 9;
int in3Pin = 10;
int in4Pin = 11;

Stepper motor(64, in1Pin, in2Pin, in3Pin, in4Pin);

void setup()
{
    pinMode(in1Pin, OUTPUT);
    pinMode(in2Pin, OUTPUT);
    pinMode(in3Pin, OUTPUT);
    pinMode(in4Pin, OUTPUT);

    // this line is for Leonardo's, it delays the serial interface
    // until the terminal window is opened
    // while (!Serial);

    Serial.begin(9600);
    motor.setSpeed(30); // 30 rpm.
```

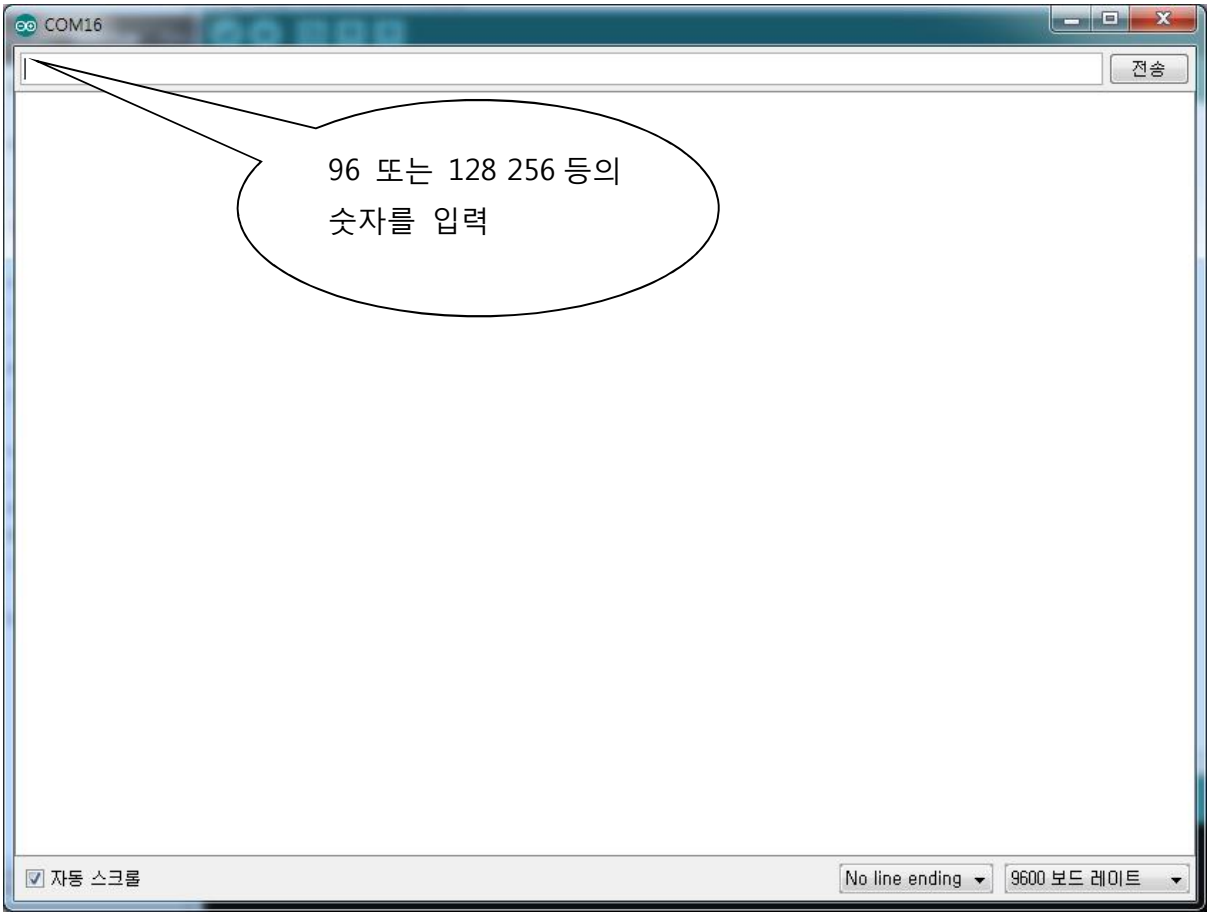
```
}

void loop()
{
    if (Serial.available())
    {
        int steps = Serial.parseInt();
        motor.step(steps);
    }
}
```

위의 코드는 시리얼로부터 정수를 받아 클래스 변수 motor 의 함수 step() 호출 해주는 예제입니다.
위의 코드를 스케치에서 업로드 후 시리얼 모니터 창을 열고 64, 128, 256 등의 숫자를 입력해 봅니다. 입력 후 스테퍼 모터 회전 확인 바랍니다.
시리얼 입력 후, 다시 2 초 3 초 후에 적당한 입력 값을 입력 하도록 합니다.



시리얼 모니터 창을 엽니다.





스테퍼 모터가 지정된 스텝만큼 회전합니다.

스테퍼 모터 스피드 지정과 스텝 회전에 대한 이해가 필요합니다.
모터 스피드는 위의 함수 motor.setSpeed(20); 사용 할 경우 함수의 파라미터 20 이라는 의미는 RPM 입니다. Revolutions Per Minute (1 분당 회전수) 입니다. 즉 1 분에 20 번 회전할 수 있는 속도를 의미합니다.

16> GREEN LED X 5

Blue light emitting diode
초록색 LED 5 개.

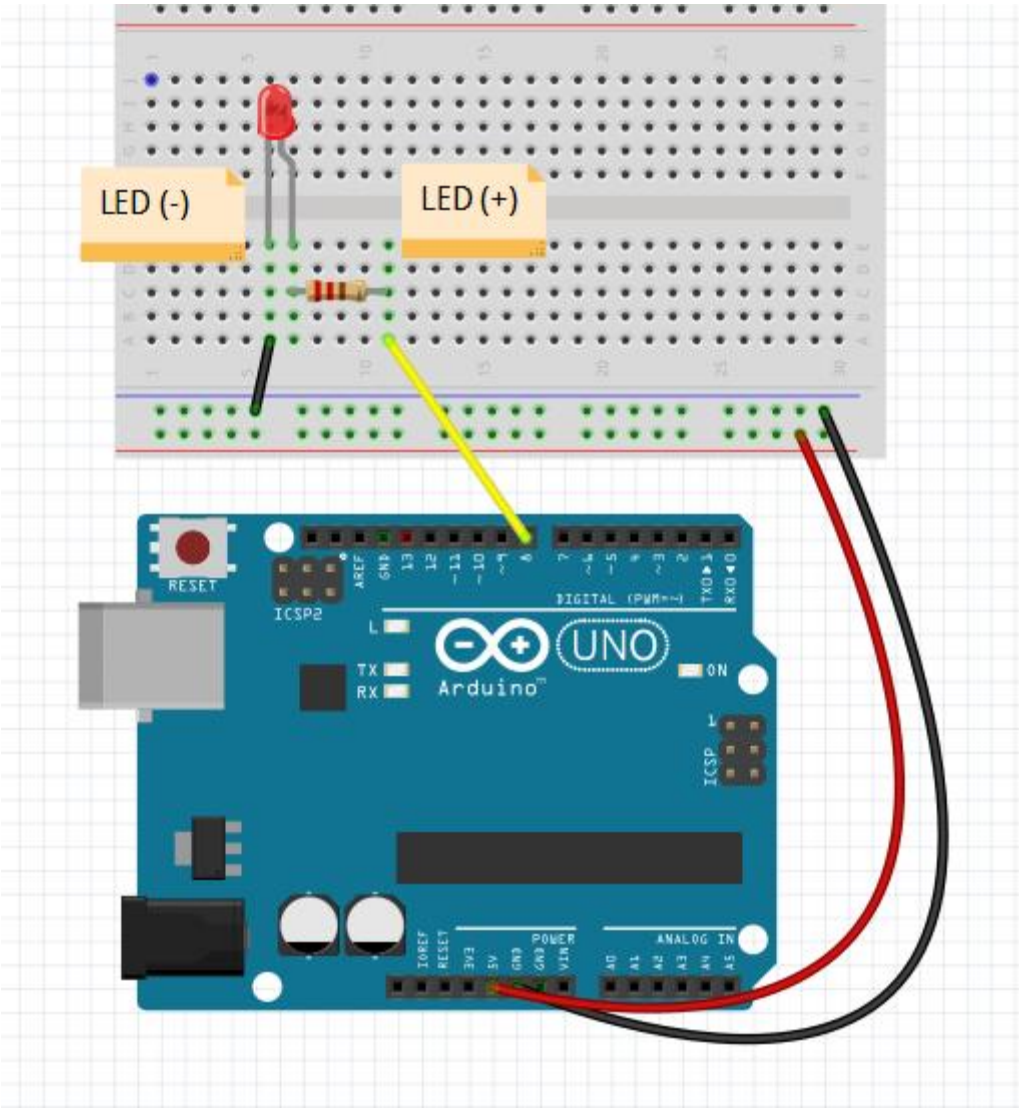
17> YELLOW LED X 5

Yellow light emitting diode
노란색 LED 5 개.

18> RED LED X 5

Red light emitting diode
빨간색 LED 5 개

아두이노에서 5V 전류 연결 후 사용시에는 LED 온/오프 시에는 220 ohm 정도의 저항을 연결합니다.



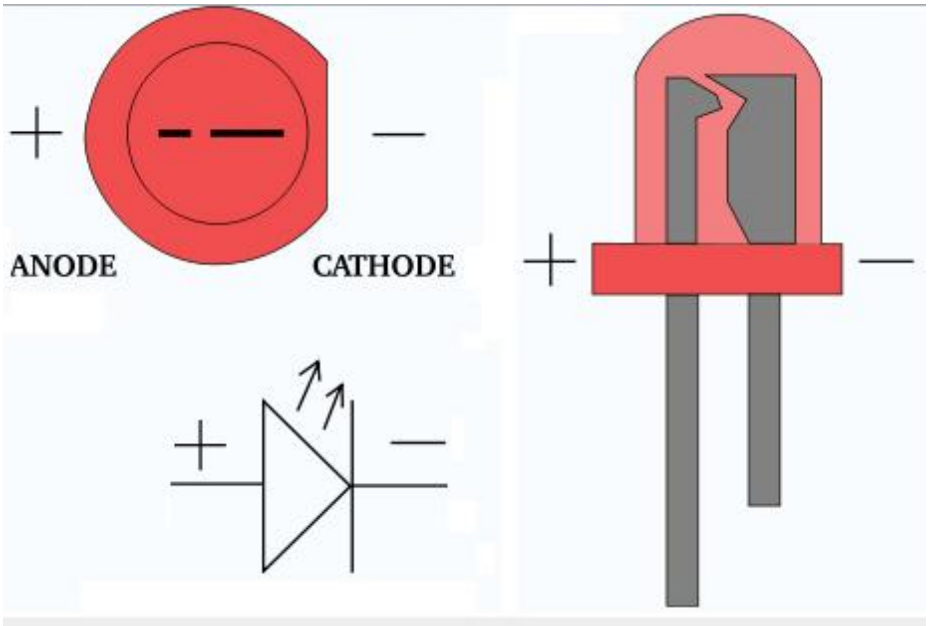
예제 코드:

```
int ledPin=8; //set IO pin of LED in control

void setup()
{
    pinMode(ledPin,OUTPUT); //set digital pin IO is OUTPUT
}

void loop()
{
    digitalWrite(ledPin,HIGH); //set PIN8 is HIGH , about 5V
```

```
    delay(1000); //delay 1000ms, 1000ms = 1s
    digitalWrite(ledPin,LOW); //set PIN8 is LOW, 0V
    delay(1000); //delay 1000ms, 1000ms = 1s
}
```

19 LED 종류와 전압 사용 참조

LED



LED 는 보통 아래와 같은 전압을 사용합니다.

색상	구 분	최소전압	최대전압	전류(일반)	전류(최대)
적색●	Red	1.8V	2.3V	20 mA	50 mA
오렌지●	Orange	2.0V	2.3V	30 mA	50 mA
황색●	Real Yellow	2.0V	2.8V	20 mA	50 mA
진한초록●	Emerald Green	1.8V	2.3V	20 mA	50 mA
초록●	Real Green	3.0V	3.6V	20 mA	50 mA
밝은 청색●	sky Blue	3.4V	3.8V	20 mA	50 mA
청색●	Real Blue	3.4V	3.8V	20 mA	50 mA

핑크●	Pink	3.4V	3.8V	20 mA	50 mA
백색○	White	3.4V	4.0V	20 mA	50 mA

20> 1K RESISTOR X 10

1K 저항 10 개

21> 10K OHM RESISTOR X 10

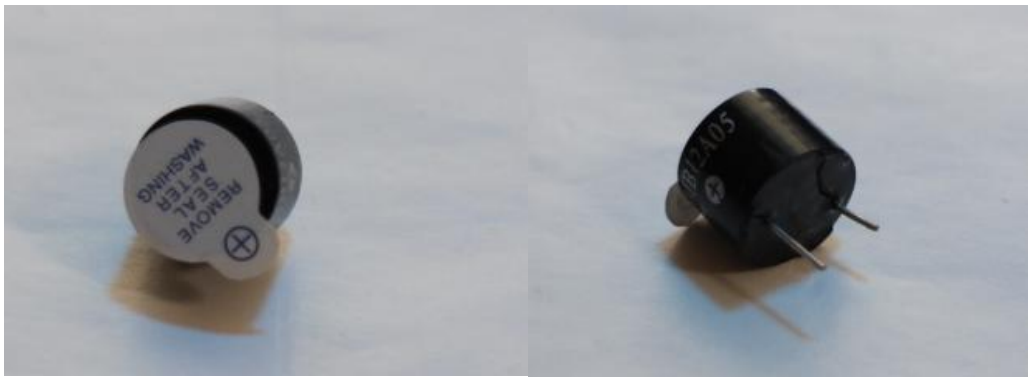
10K 저항 10 개

22> 220 OHM RESISTOR X 10

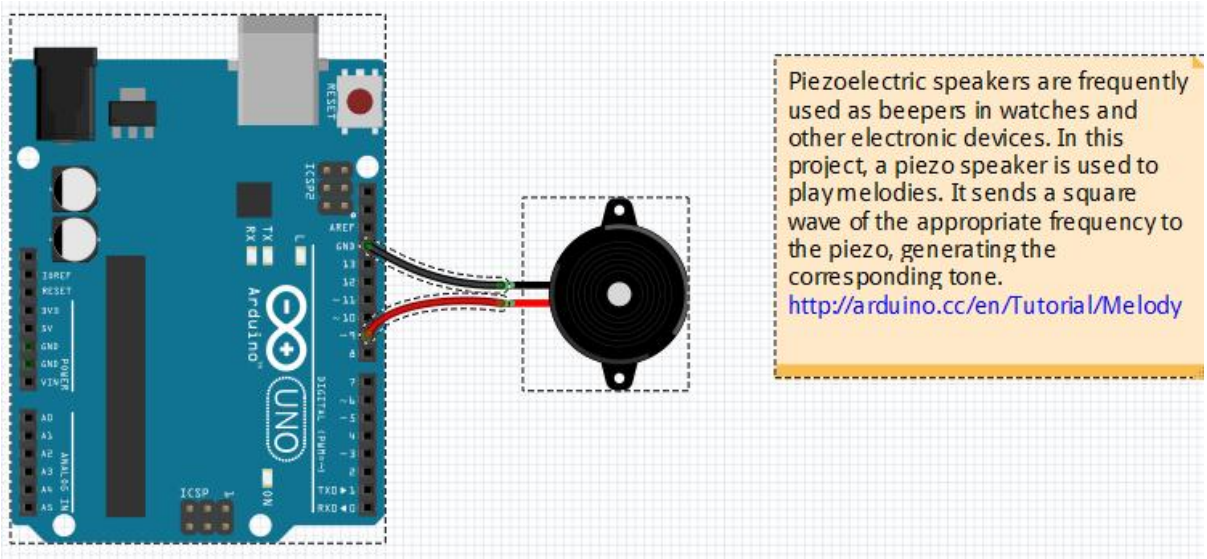
220 ohm 저항 10 개

23> PASSIVE BUZZER X 1

수동 부저 입니다.
경고, 간단한 멜로디 등의 소리를 낼 수 있는 모듈입니다. 5V 사용됩니다.



아두이노 와이어링 다이어그램



부저 하단부의 2 개의 리드선 극성을 구분하여 연결하여 줍니다.

(+) 표시된 리드선과 나머지는 (-) 리드선입니다.

아두이노 PlayTone 예제 코드
ABCDEFG 알파벳 송입니다.

```
int speakerPin = 9;

int length = 15; // the number of notes
char notes[] = "ccggaagffeeddc "; // a space represents a rest
int beats[] = { 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 4 };
int tempo = 300;

void playTone(int tone, int duration)
{
    for (long i = 0; i < duration * 1000L; i += tone * 2)
    {
        digitalWrite(speakerPin, HIGH);
        delayMicroseconds(tone);
        digitalWrite(speakerPin, LOW);
        delayMicroseconds(tone);
    }
}

void playNote(char note, int duration)
{
    char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };
    int tones[] = { 1915, 1700, 1519, 1432, 1275, 1136, 1014, 956 };

    // play the tone corresponding to the note name
    for (int i = 0; i < 8; i++)
    {
        if (names[i] == note)
        {
            playTone(tones[i], duration);
        }
    }
```

```
}

void setup()
{
    pinMode(speakerPin, OUTPUT);
}

void loop()
{
    for (int i = 0; i < length; i++)
    {
        if (notes[i] == ' ')
        {
            delay(beats[i] * tempo); // rest
        }
        else
        {
            playNote(notes[i], beats[i] * tempo);
        }
        // pause between notes
        delay(tempo / 2);
    } // end for loop
}
```

24> PIEZO BUZZER X 1

피에조 부저입니다. 5V 사용됩니다. 피에조 뜻은 압박이라는 의미입니다.
부저 하단부의 2 개의 리드선 극성을 구분하여 연결하여 줍니다.
(+) 표시된 리드선과 나머지는 (-) 리드선입니다.
작동 방법은 + 리드선에 High 일 경우에는 삑~ 하는 소리, Low 일 경우에는 소리가 안 납니다.

부저 리드핀	아두이노 보드
(−)	GND
(+)	D11

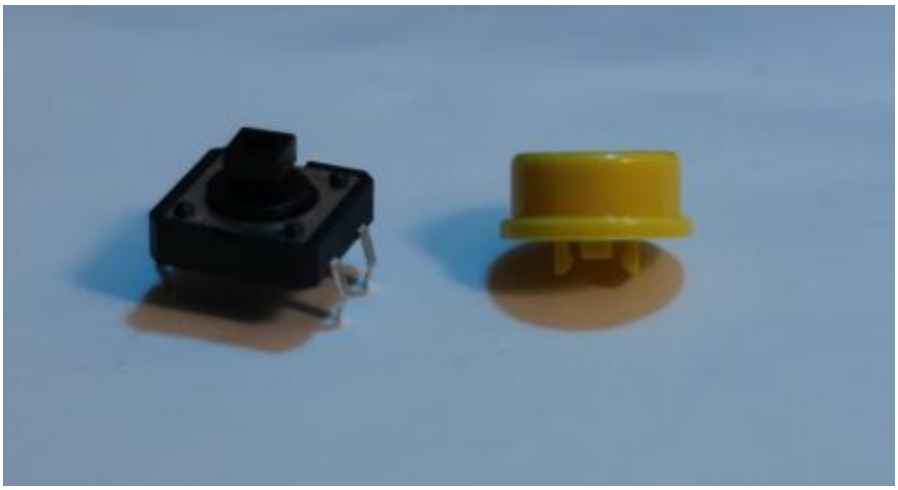
아두이노 스케치 예제코드:

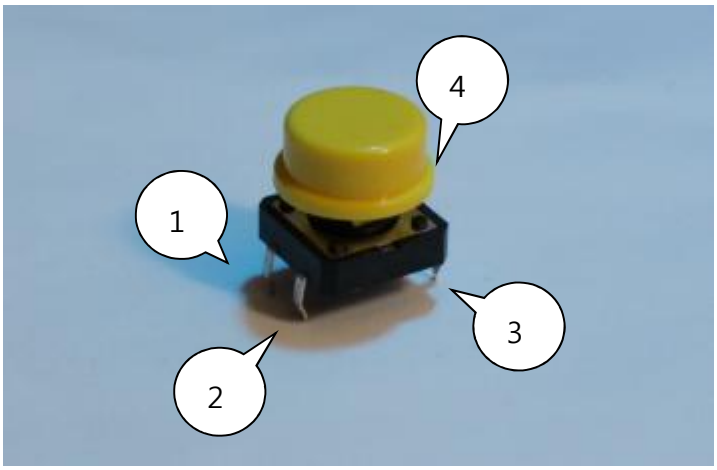
```
int buzzer=11; // 디지털 포트 11 번 지정.

void setup() {
  pinMode(buzzer, OUTPUT);
}

void loop() {
  unsigned char i,j;
  while(1) {
    for (i=0; i<80; i++) { //output sound of one frequency
```

```
        digitalWrite(buzzer, HIGH); //make a sound
        delay(1); //delay 1ms
        digitalWrite(buzzer ,LOW); //silient
        delay(1); //delay 1ms
      }
      for(i=0; i<500 ;i++) { //output sound of another frequency
        digitalWrite(buzzer, HIGH); //make a sound
        delay(2); //delay 2ms
        digitalWrite(buzzer, LOW); //silient
        delay(2);   //delay
      }
    }
}
```

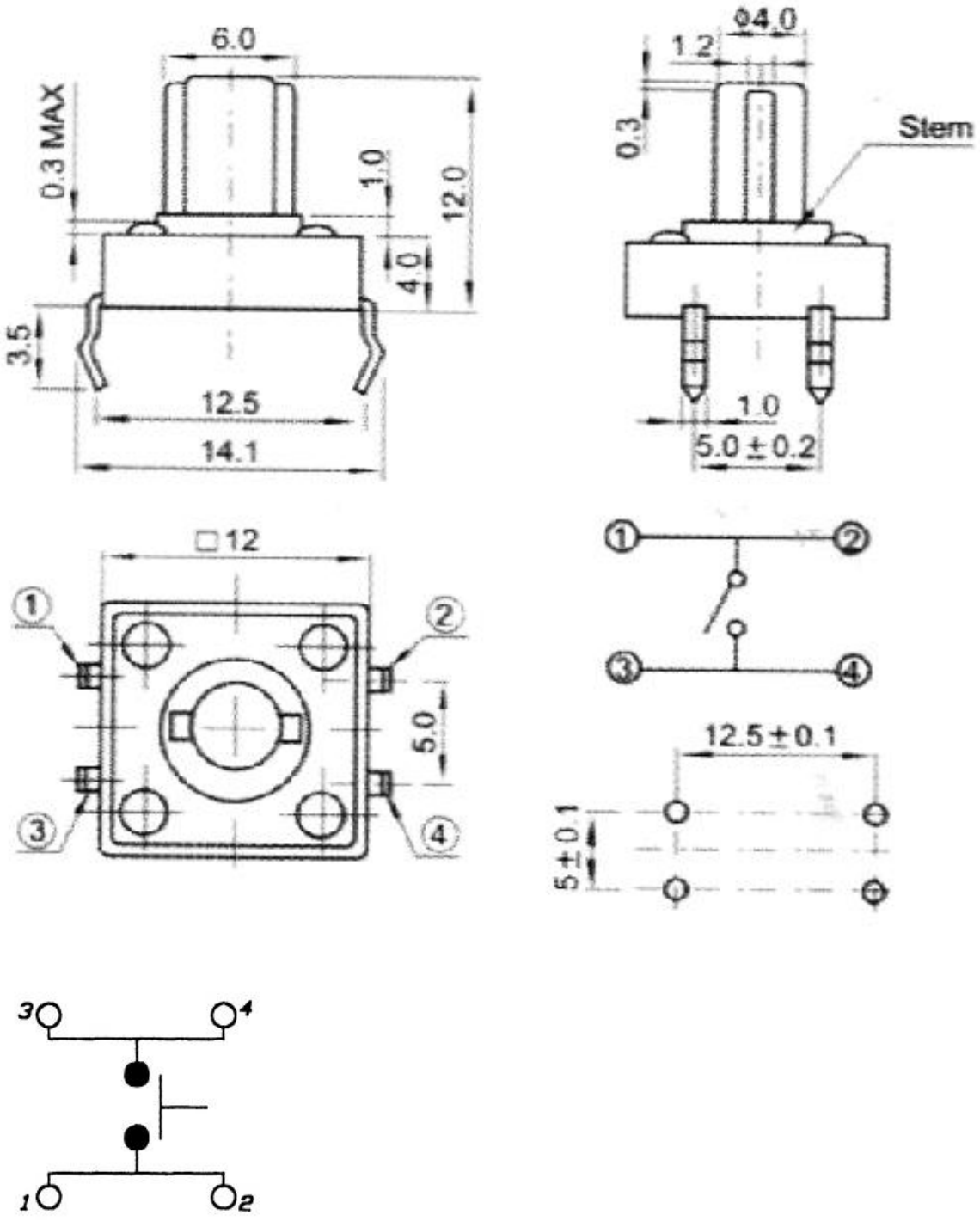
25> KEY MODULE (WITH HAT) X 4

Tact Switch 12x12 mm 입니다.
Tact 는 Tactile 의 약자입니다. 접촉 스위치, 접촉 버튼으로 번역할 수 있습니다..
노란색 캡이 있어서 버튼 키 위에 덮어 주면 완성 됩니다.
물론 노란색 캡이 없어도 버튼 기능은 됩니다.





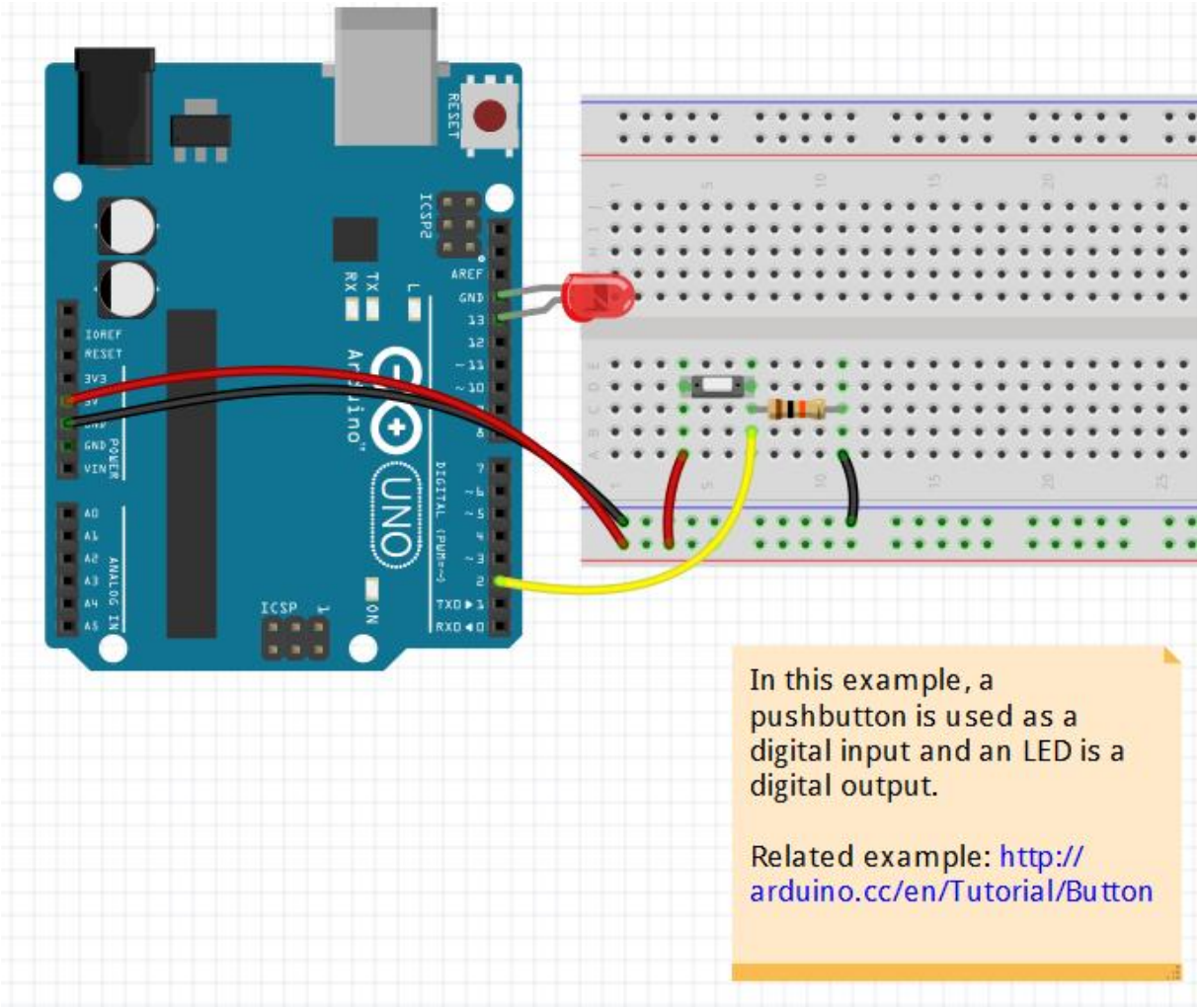
Tact Switch (or Tact Button)은 여러 용도로 사용 되고 있습니다.
기초 전기/전자 부품이면서, 크기, 형태만 약간 다를 뿐 수많은 가전/산업분야의 전자 제품, 전자 기기에 많이 사용됩니다.
버튼의 내부는 아래와 같은 구조입니다.



아래의 예제는 버튼이 눌리면 LED 켜는 예제입니다.

Pull Down 예제입니다.

예제코드는 버튼이 눌려진 상태인 경우에만 13 번 포트에 연결된 LED 켜는 예제입니다.




```
// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = 2; // the number of the pushbutton pin
const int ledPin = 13; // the number of the LED pin

// variables will change:
int buttonState = 0; // variable for reading the pushbutton status

void setup() {
      // initialize the LED pin as an output:
      pinMode(ledPin, OUTPUT);
```

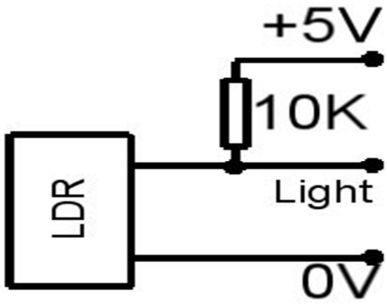
```
    // initialize the pushbutton pin as an input:
    pinMode(buttonPin, INPUT);
}

void loop(){
    // read the state of the pushbutton value:
    buttonState = digitalRead(buttonPin);

    // check if the pushbutton is pressed.
    // if it is, the buttonState is HIGH:
    if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
    }
    else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
    }
}
```

26> TILT SWITCH X 2

기울기 센서 스위치 SW-520D 모듈입니다.

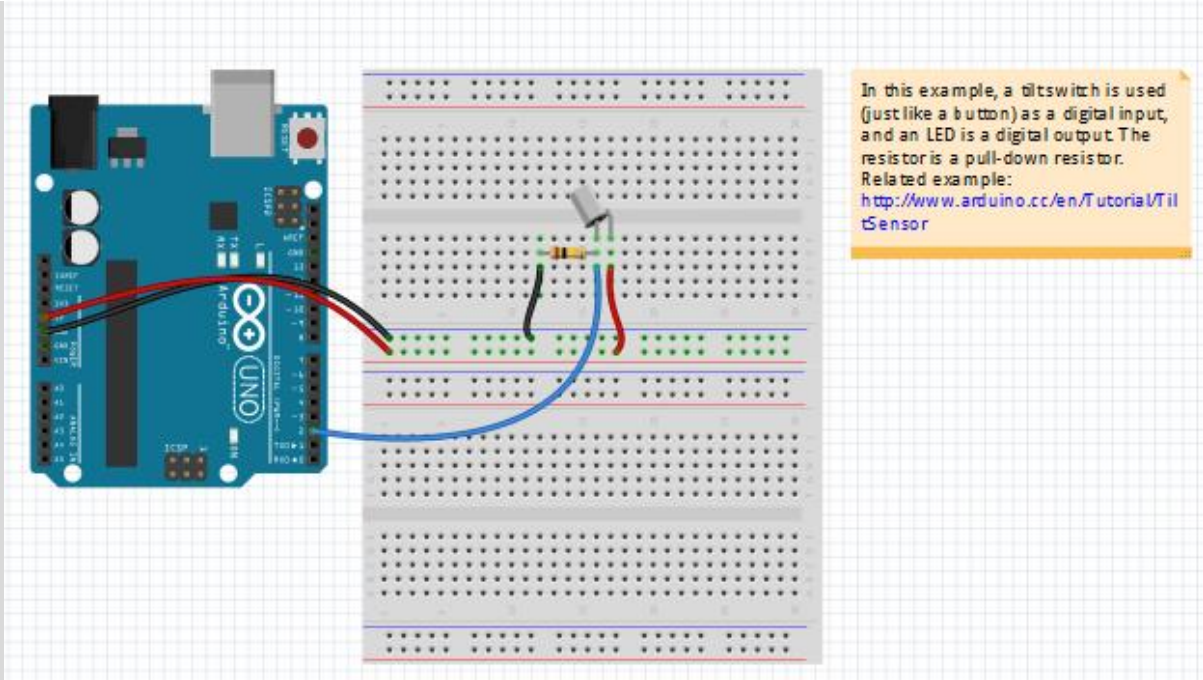
기울기 센서입니다. 볼 스위치라고도 합니다.

위의 예시 이미지를 보시면 내부에 2 개의 볼이 들어 있습니다. 기울기 또는 진동 등에 의한 스위치 역할을 합니다.
본 예제에서는 기울기 값에 의한 단순한 스위치 On / Off 합니다.
회로 개념도는 아래와 같습니다.



SW-520D 센서는 2 개의 핀에 GND, SIGNAL 구분은 없습니다. 신호를 받을 핀에는 10K 저항을 연결 해 아두이노 보드 디지털 2 번 포트에 연결합니다.

아두이노 보드로의 연결은 아래 이미지를 참조 합니다.



스케치 예제 코드는 Digital->Button 예제를 그대로 사용 합니다.

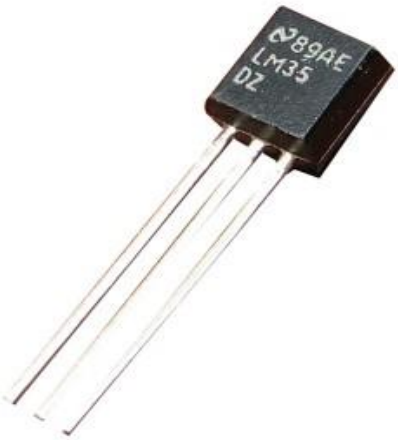
```
// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = 2;     // the number of the pushbutton pin
const int ledPin =  13;      // the number of the LED pin

// variables will change:
int buttonState = 0;         // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}

void loop() {
```

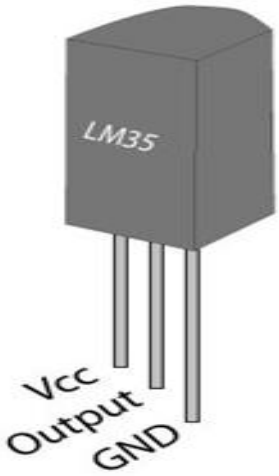
```
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  }
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

아두이노에 버튼 예제 프로그램 업로드 후 센서의 기울기를 조절 해 봅니다.
또는 센서를 툭툭 쳐서 건드려 봅니다.
LED 13 번 온/오프 되는 것을 확인 합니다.
실제 제품에 적용 시에는 순간의 기울기나 진동을 체크하긴 보단 타이머 인터럽트로 일정 시간 센서 값 평균을 구해서, 변동이 있는 경우만 처리하는 것도 좋습니다.
특정한 목적, 특수한 경우가 아닌 경우, 거의 모든 아날로그 센서 값 기준 하드웨어 동작은 지정 시간 대비 또는 지정 클럭 수 대비 평균을 구하여 프로그래밍 하도록 합니다.

27> LM35 SENSOR MODULE X 1

온도 센서 입니다. LM35 DZ 센서입니다.



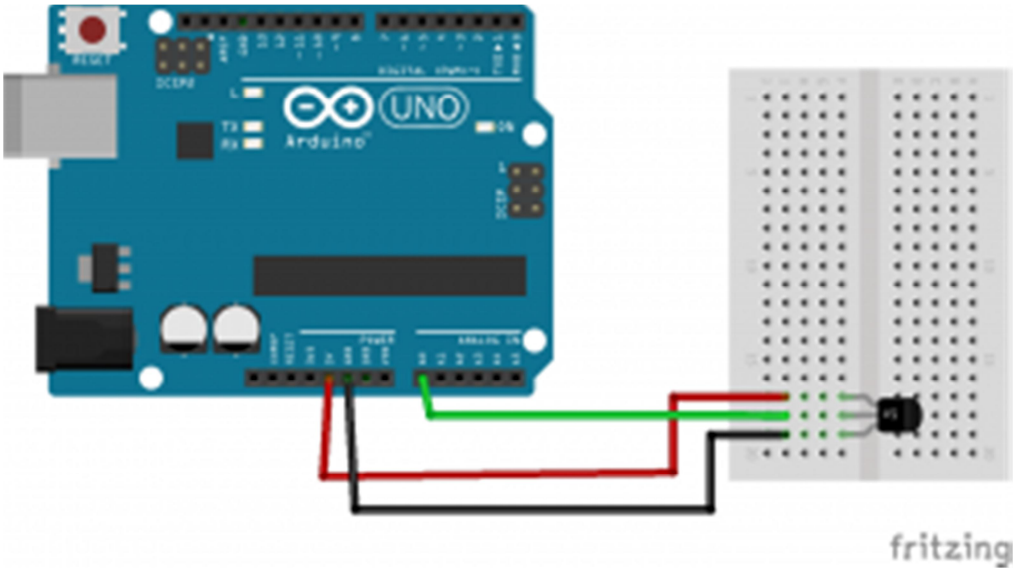


섭씨 1 도에 10 mA 의 변화 값을 구할 수 있는 온도 센서입니다.
측정 온도의 범위는 섭씨온도 0~100 도입니다.

아두이노 연결:

Pin	Function	Name
1	Supply voltage; 5V (+35V to -2V)	VCC
2	Output voltage (+6V to -1V)	Output
3	Ground (0V)	Ground

아두이노 와이어링 다이어그램입니다.



온도 값 계산시 입력 전원 5V, 3V3 참조 기준 연산 값으로 계산 해주면 됩니다.
예제코드:

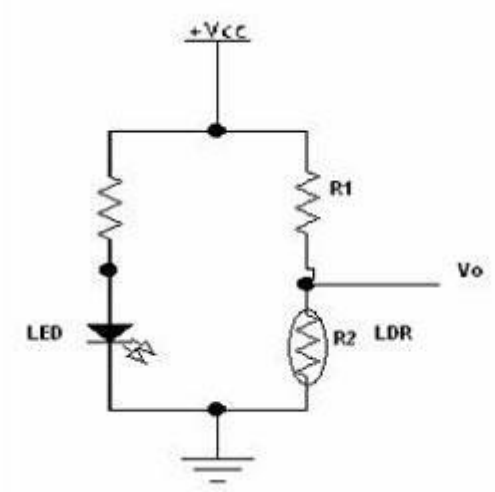
```
// 1 초마다 온도를 구합니다.
const unsigned int TEMP_SENSOR_PIN = 0;    // 온도 센서 입력 아날로그 포트
const float SUPPLY_VOLTAGE = 5.0;          // 5V 전류 연결 정의 합니다.
const unsigned int BAUD_RATE = 9600; //

const float get_temperature() {
  const int sensor_voltage = analogRead(TEMP_SENSOR_PIN);
  const float voltage = sensor_voltage * SUPPLY_VOLTAGE / 1024;
  return (voltage * 100);
}

void setup() {
  Serial.begin(BAUD_RATE); // 보 레이트 설정.
}

void loop() {
```

```
  Serial.print(get_temperature());
  Serial.println(" C");
  delay(1000);
}
```

28> PHOTO RESISTOR X 3

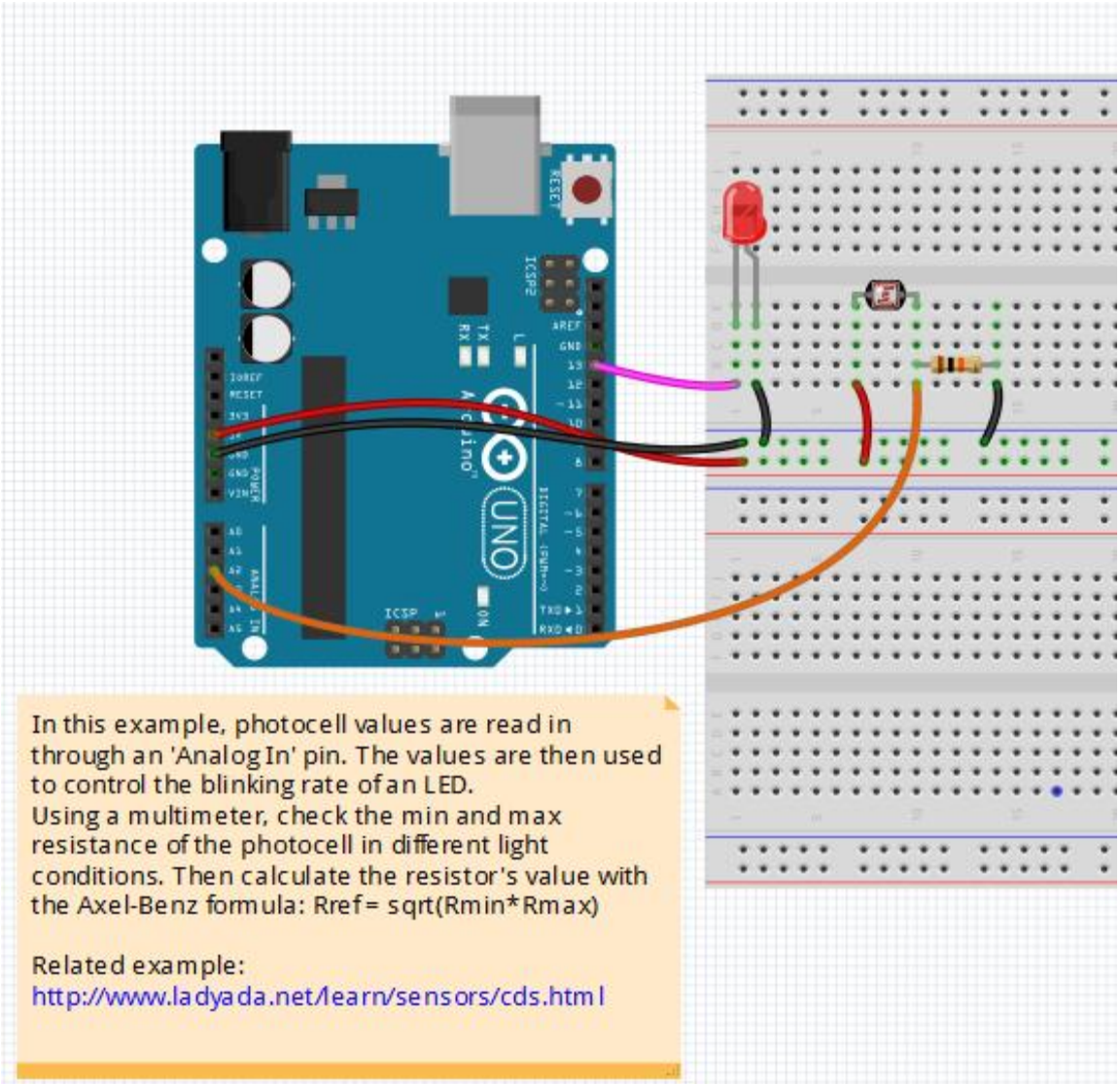
광 센서 입니다.

조도 센서 / 라이트 센서 라는 명칭도 있습니다.
다른 용어로는 LDR 입니다. Light Dependent Resistor



광소자, 광도전 센서라고도 합니다.
광(Photo) + Resistor (저항) 합성어입니다.
외부 빛의 세기에 따라 내부 저항 값이 감소합니다. 즉, 가변 저항인데 빛의 세기에 따라 자동으로 변화한다고 보면 됩니다.
광센서는 저항 타입이라 양극(+), 음극(-) 구분하지 않습니다.
외부 빛이 적거나 차단되면 저항의 크기가 커집니다. 반대로 외부 빛이 많다면 광센서의 저항 매우 작아 집니다.

아두이노에 와이어링은 아날로그 포트, +5V 에 10K 저항을 적용하여 테스트 해봅니다.
풀 다운입니다.



키트에 3 개 센서 포함된 이유는 빛 자체가 방향성 데이터이므로, 가장 이상적인 빛 센서 체크는 삼각형 기준 밖으로(또는 상향) 센서를 배치 후, 각 3 개의 센서 입력의 평균값으로 조도 값을 측정하면 매우 정확한 결과치를 얻을 수 있습니다.
물론 1 개로도 충분합니다.
회로 C&R 구성에 따라 가장 밝은 빛에서의 체크 용도, 보통 날씨의 조도, 빛이 약간 있는 어두운 곳에서의 사용 등등 많이 구분됩니다.

```
int lightPin = 3;           // 아날로그 포트 A02
int threshold = 250;        // 조도값 250 기준으로 LED 온/오프

void setup()
```

```
{
      Serial.begin(9600);   //Begin serial communcation
      pinMode(13, OUTPUT);
}

void loop()
{
      Serial.println(analogRead(lightPin));

      if(analogRead(lightPin) > threshold )
      {
            digitalWrite(13, HIGH);
            Serial.println("high");
      }else{
            digitalWrite(13, LOW);
            Serial.println("low");
      }

      delay(100);
}
```

29> FIRE X 1 (FLAME SENSOR)

Flame Sensor 입니다. 화염, 불꽃 감지 센서입니다.

명칭 그대로 불꽃 감지 센서입니다.
화염의 고유 760nm~1100nm 파장의 빛을 검출하는데 사용 됩니다.

화염 센서 DataSheet:
http://www.igameplus.com/pds/gpshop/arduino/pdf/flamesensor.pdf

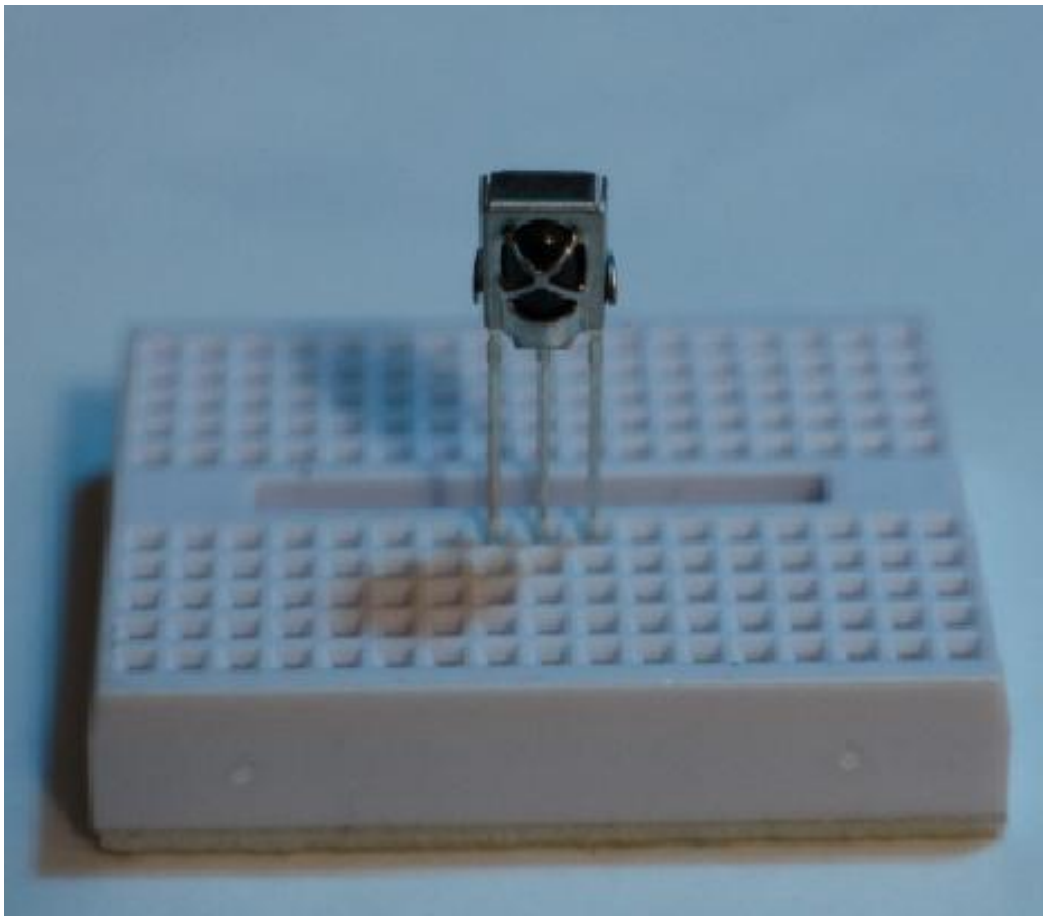
스케치 예제코드

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  int sensorValue = analogRead(A0);
  Serial.println(sensorValue);
}
```

30> INFRARED RECEIVER X 1

적외선 수신기입니다.
적외선 리모컨 컨트롤러 또는 적외선 송신기 에서 보내는 신호를 받는 모듈입니다.

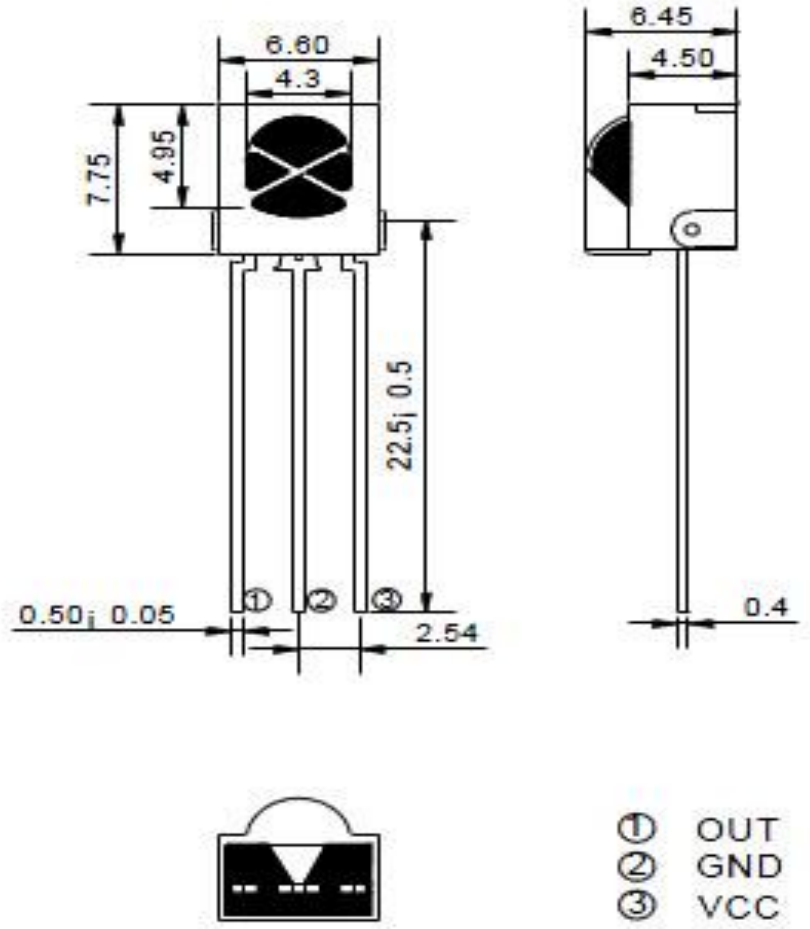
30.1 수신기 부품 형태



리모트 컨트롤러

리모트 컨트롤러의 배터리 장착 이미지입니다.

IR Receiver 핀맵입니다.

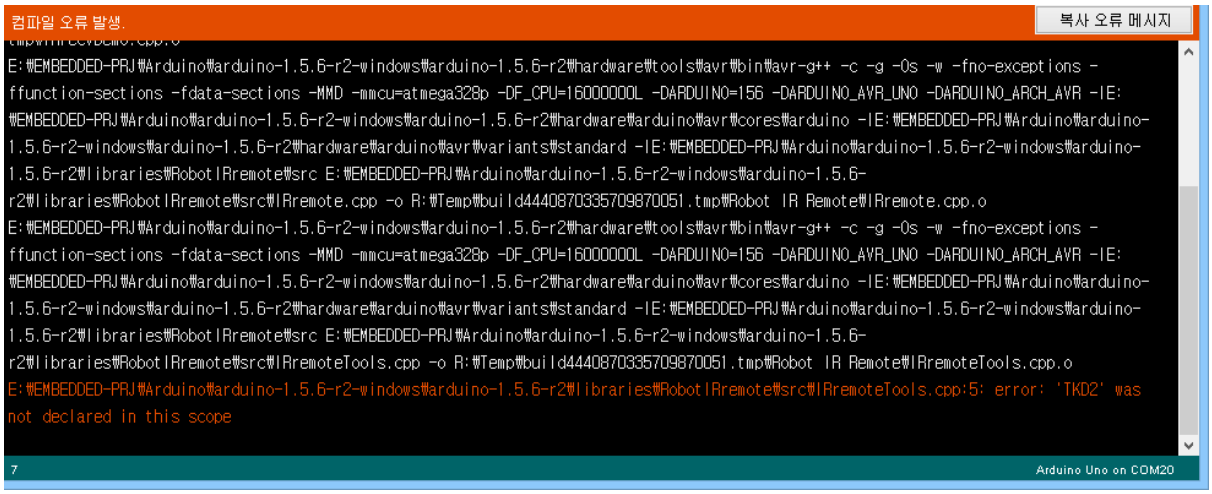


적외선 리시버	아두이노 우노
OUT	11
GND	GND
VS	5V

리모트 컨트롤 라이브러리 & 예제 코드입니다.
적외선 컨트롤러 모든 참조를 할 수 있습니다.
https://github.com/shirriff/Arduino-IRremote
or 다운로드 Arduino-IRremote-master.zip

위의 라이브러리를 스케치 설치 후 아래의 예제를 테스트 가능합니다.
IRremote 에 포함된 수신기 예제 코드입니다.

>> 예제 빌드시 아래와 비슷한 컴파일 에러 메시지가 나올 수 있습니다.



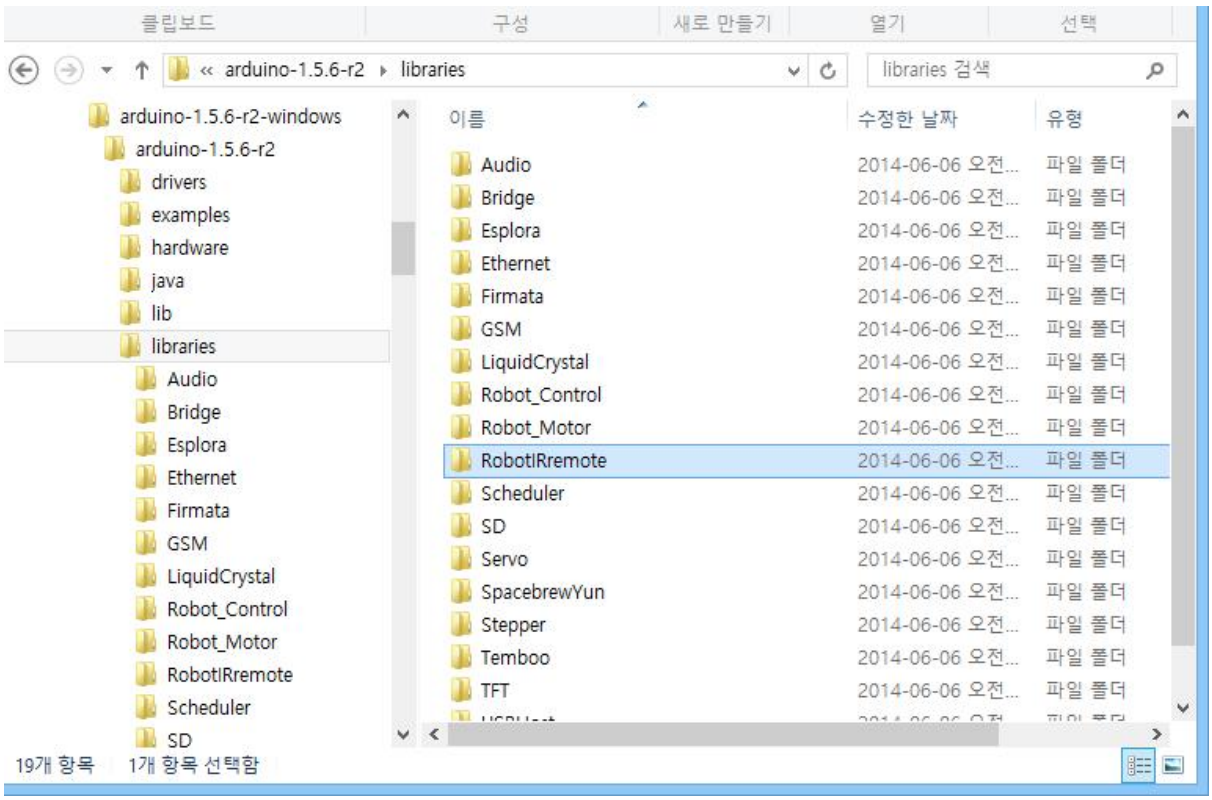
메시지의 내용은

".....Arduino₩arduino-1.5.6-r2-windows₩arduino-1.5.6-r2₩libraries₩RobotIRremote₩src₩IRremoteTools.cpp:5: error: 'TKD2' was not declared in this scope

IRRemote.h 라는 라이브러리 헤더 파일 명칭이 아두이노의 로봇 IR 과 중복되어 RobottIRRemote 스케치 기본 라이브러리가 빌드시 나오는 현상입니다.

RobotIRRemote 라이브러리의 헤더파일 중복을 피하기 위해 RobotIRRemote 디렉터리를 옮기거나 삭제를 합니다.

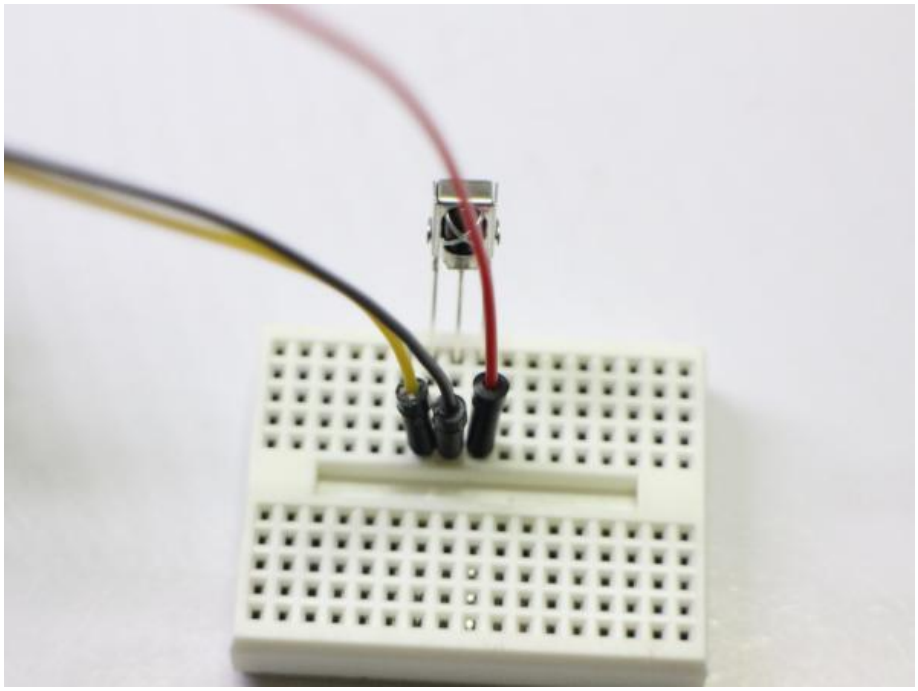
윈도우 탐색기를 열어서 아두이노 스케치 프로그램이 있는 디렉터리로 갑니다.



아두이노의 기본 라이브러리 디렉터리 → RobotIRRemove 라는 디렉터리를 다른 곳으로 옮겨 놓습니다.

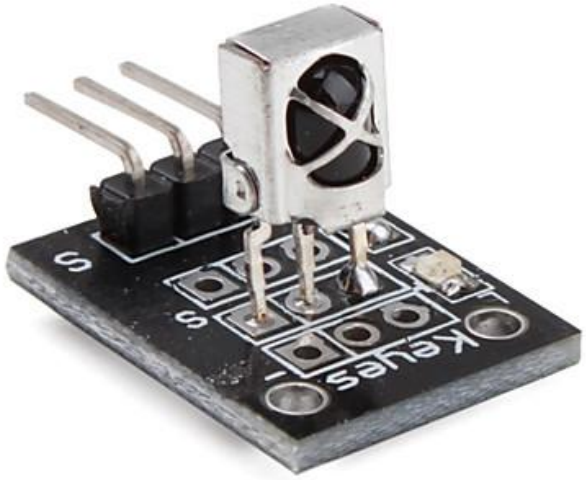
아래의 예제 코드를 열어서 빌드 해봅니다.

IR 수신기의 신호 포트와 아두이노 우노의 PWM 포트와 연결하면 됩니다.

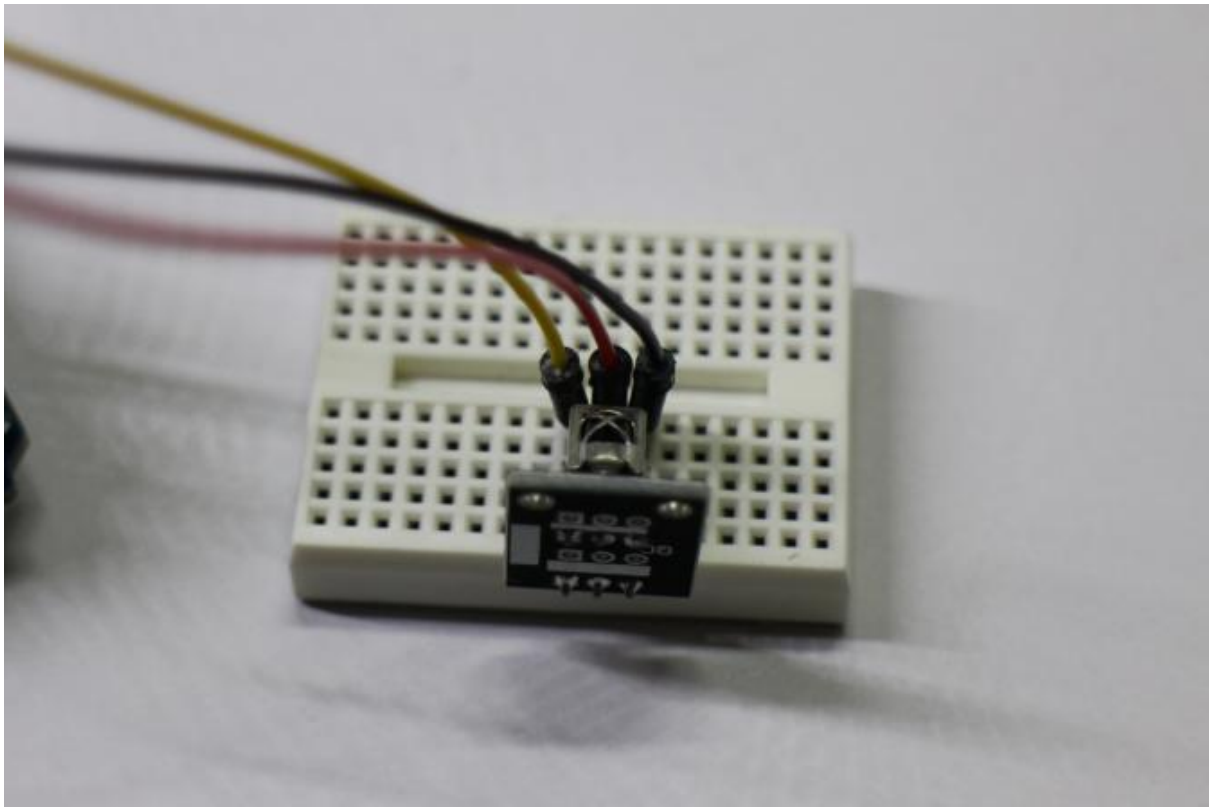


30.2 IR 수신 모듈 와이어링

IR 수신 모듈 IR 수신기의 신호 포트와 아두이노 우노의 PWM 포트와 연결하면 됩니다. 수신기 본체와 브레이크아웃보드의 와이어링 순서가 조금 틀립니다.



브레드보드에 와이어링 이미지입니다. 연결 와이어링의 색상을 참조하기 바랍니다.



IR 수신 예제 코드.

```
/*
IR_remote_tester_and_detector
Connect the output pin of Infrared remote to DIG 2 or 11
Connect an LED to pin 13.
*/
#include <IRremote.h>

const int irReceiverPin = 11; // PWM
const int ledPin = 13;
IRrecv irrecv(irReceiverPin); //create an IRrecv object
decode_results decodedSignal; //stores results from IR sensor

void setup()
{
```

```
    pinMode(ledPin, OUTPUT);
    irrecv.enableIRIn(); // Start the receiver object
}

boolean lightState = false; //keep track of whether the LED is on
unsigned long last = millis(); //remember when we last received an IRmessage

void loop()
 {
    //this is true if a message has been received
    if (irrecv.decode(&decodedSignal) == true)
    {
        if (millis() - last > 250)
        {
            //has it been 1/4 sec since last message
            lightState = !lightState; //toggle the LED
            digitalWrite(ledPin, lightState);
        }

        last = millis();
        irrecv.resume(); // watch out for another message
    }
}
```

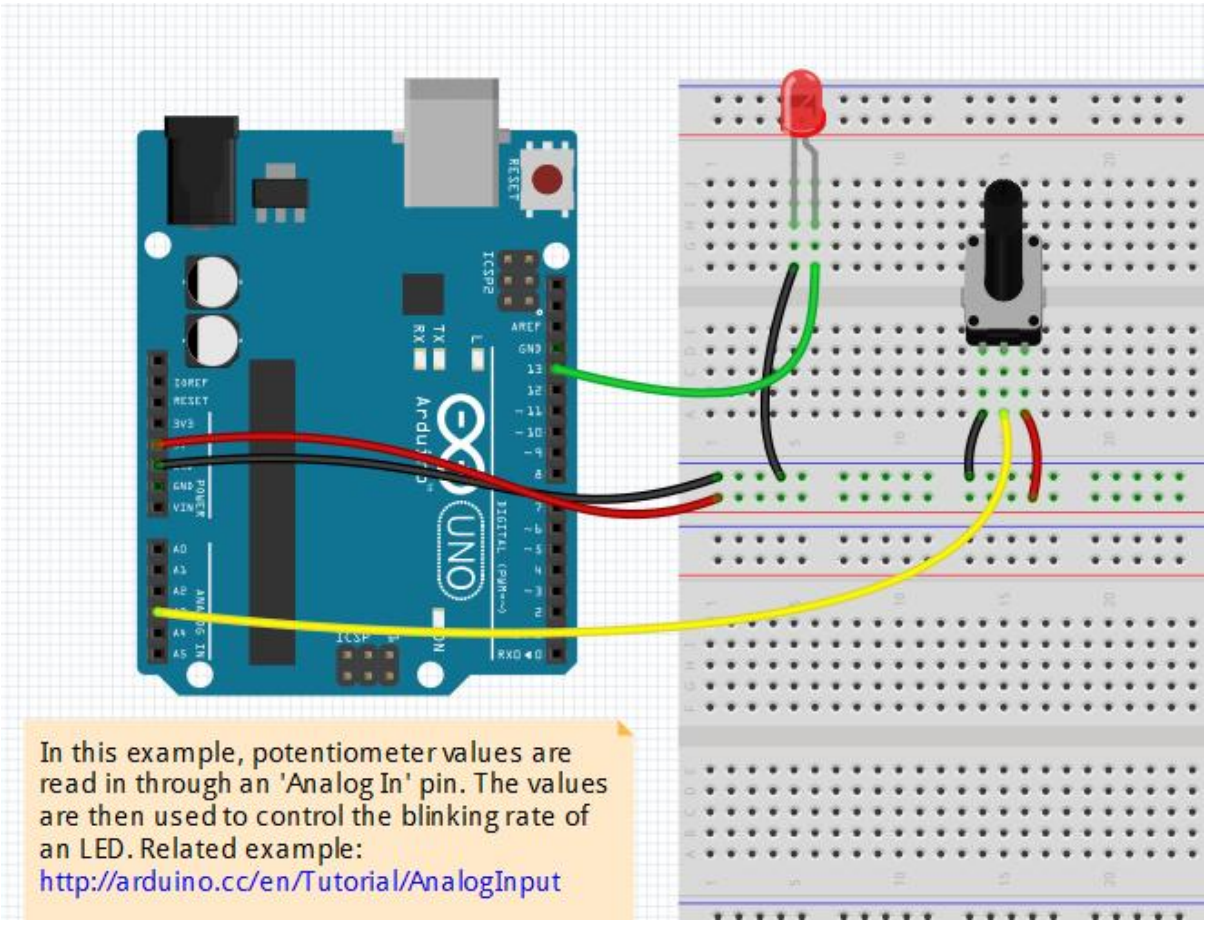
31> ADJUSTABLE POTENTIOMETER X 1

가변저항 10K 입니다.

어저스터블 포텐쇼미터 10K / 조절 가능한 전위차계 입니다.

가변저항으로 LED 밝기 조절하는 예제입니다.

아두이노 와이어링 다이어그램



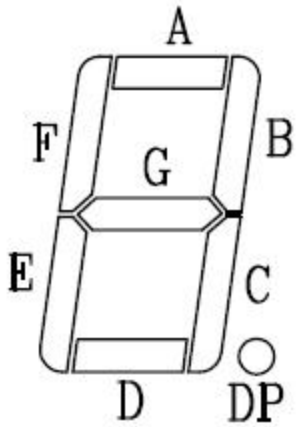
아래의 코드는 analogRead(), analogWrite() 함수의 사용 예제입니다.

가변저항 10K 의 손잡이를 돌려서 9 번 포트에 연결된 LED 의 밝기 조절하는 예제 코드입니다.

```
// analogRead values go from 0 to 1023,
// LED connected to digital pin 9
// LED 를 아두이노 보드 13 번에 연결.
int ledPin = 13;

// potentiometer connected to analog pin 3
// 가변저항 10K 의 신호선을 아두이노 보드 3 번 핀에 연결
int analogPin = A3;

int val = 0;            // variable to store the read value

void setup()
{
    pinMode(ledPin, OUTPUT);    // sets the pin as output
}

void loop()
{
    // read the input pin
    val = analogRead(analogPin);

    // analogRead values go from 0 to 1023, analogWrite values from 0 to 255
    analogWrite(ledPin, val / 4);
}
```

1-Digit 7 Segment 모듈입니다.

숫자 하나당, 7 개 내부 LED 를 On/Off 시켜 표시하는 장치 또는 모듈입니다.
1-Digit 7 Segment 구성 인덱스



추가 1 개의 LED 는 DOT 표시로 사용 됩니다. 결국 LED 8 개로 구성 되어 있지만 숫자 표시는 7 개 LED 사용 되므로 보통 7-segment 라고 부릅니다.

FND (Flexible Numeric Display) 라고 부릅니다.
마이크로 컨트롤러 교재, 문서 또는 최근엔 아두이노를 배울 때 꼭 거쳐야 되는 부분이 FND 모듈입니다.

FND 데이터쉬트 또는 설명 다이어그램을 보면 항상 COM 이라는 부분이 보입니다.
COM 이라는 것은 COMMON, 즉 공통 핀 개념으로 내부 LED 의 구조에 따라 GND, 또는 VCC 가 될 수 도 있습니다.
사용시 데이터 쉬트를 참조해야 하는 부분입니다.
COMMON 핀이 VCC 인 경우에는 커먼 애노드 타입, COMMON 핀이 GND 경우에는 커먼 캐소드라고 합니다.

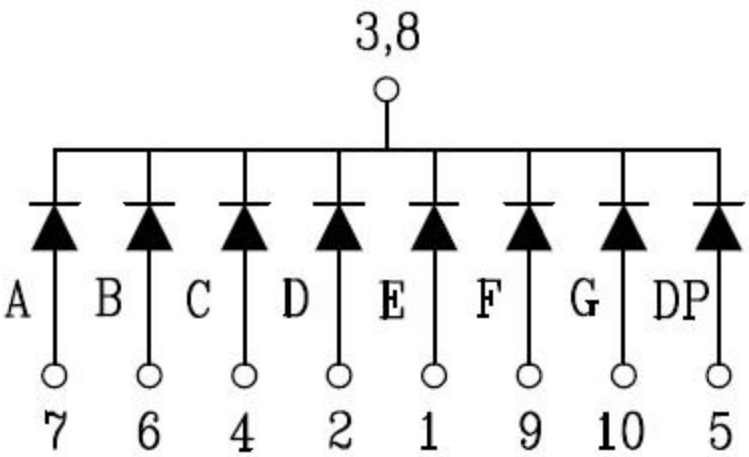
애노드, 캐소드는 꼭 알아야 할 단어이기도 합니다.

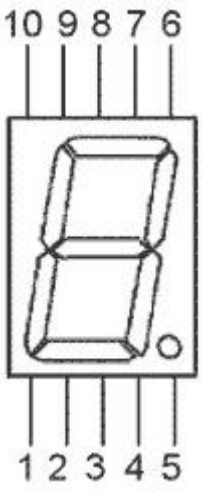
이미 아시는 단어이겠지만, Anode 를 **(+)**, Cathode 는 **(-)** 입니다.

즉 단어 뜻을 이해하신다면 FND 이해가 쉽게 되리라 봅니다.
캐소드 - com7-Segment 는 양극 입력 시 발광하는 캐소드(Cathode) 타입과 음극 입력 시 발광하는 애노드(Anode), 본 실습에는 캐소드 타입을 이용 합니다

애노드 타입: 공통 핀 + 이므로 나머지 LED 는 – 연결 후 On/Off
캐소드 타입: 공통 핀 – 이므로 나머지 LED 는 + 연결 후 On/Off

Common Cathode Circuit

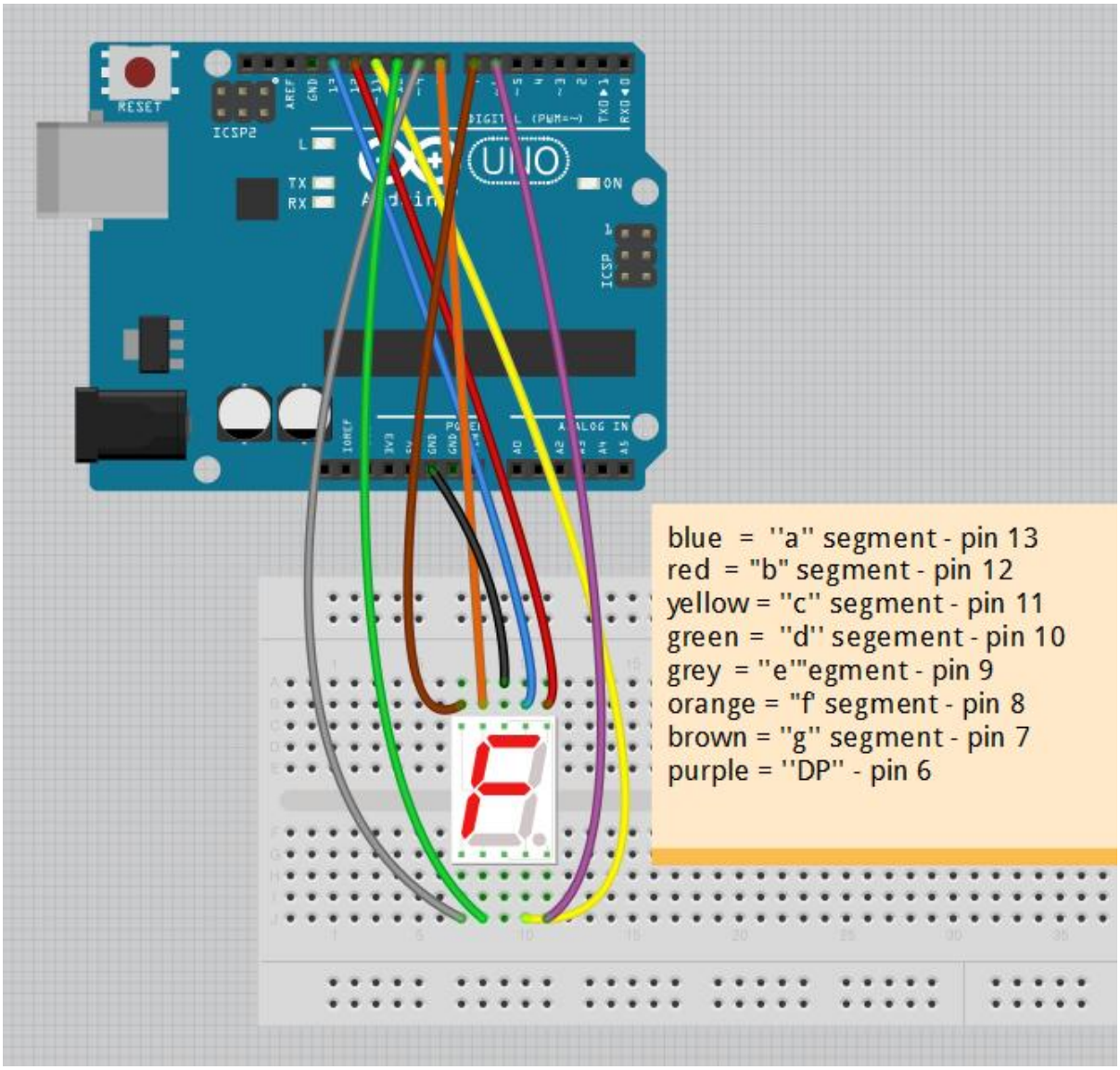


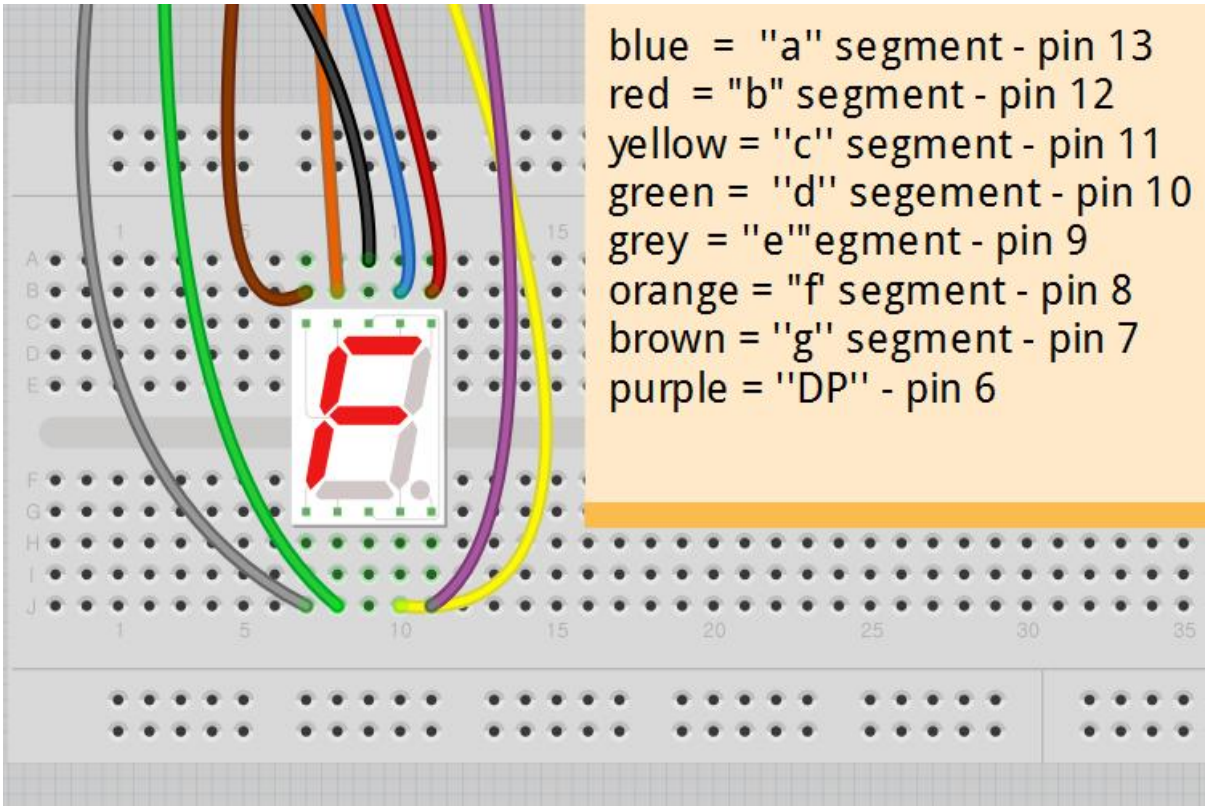


DataSheet:

http://www.igameplus.com/pds/gpshop/arduino/pdf/7-segment-2I17KS85.pdf

아래는 아두이노 우노에서 커먼 캐소드 FND 모듈 와이어링 설명 이미지 입니다.





아두이노 스케치 예제 코드입니다. (게시판에도 있습니다)

```
//
// 시리얼 포트로 번호를 입력 받아 FND 에 숫자를 표시하는 예제 입니다.
// 연결 핀을 정의합니다.
//
int a = 13;
int b = 12;
int c = 11;
int d = 10;
int e = 9;
int f = 8;
int g = 7;
int dp = 6;

void setup() {
Serial.begin(9600); // uart0 속도 지정.
```

```
// 사용되는 포트 방향을 모두 OUTPUT 으로 합니다.
  pinMode(a , OUTPUT);
  pinMode(b , OUTPUT);
  pinMode(c , OUTPUT);
  pinMode(d , OUTPUT);
  pinMode(e , OUTPUT);
  pinMode(f , OUTPUT);
  pinMode(g , OUTPUT);
  pinMode(dp , OUTPUT);
}


void loop(){
 int val = Serial.read() - '0';
 if (val == 0){
    digitalWrite(a , HIGH);
    digitalWrite(b , HIGH);
    digitalWrite(c , HIGH);
    digitalWrite(d , HIGH);
    digitalWrite(e , HIGH);
    digitalWrite(f , HIGH);
    digitalWrite(dp , HIGH);
    delay(1000);
    digitalWrite(a , LOW);
    digitalWrite(b , LOW);
    digitalWrite(c , LOW);
    digitalWrite(d , LOW);
    digitalWrite(e , LOW);
    digitalWrite(f , LOW);
    digitalWrite(dp , LOW);
 }
 if (val == 1){
    digitalWrite(b , HIGH);
```

```
    digitalWrite(c , HIGH);
    digitalWrite(dp , HIGH);
    delay(1000);
    digitalWrite(b , LOW);
    digitalWrite(c , LOW);
    digitalWrite(dp , LOW);
  }
  if (val == 2){
    digitalWrite(a , HIGH);
    digitalWrite(b , HIGH);
    digitalWrite(d , HIGH);
    digitalWrite(e , HIGH);
    digitalWrite(g , HIGH);
    digitalWrite(dp , HIGH);
    delay(1000);
    digitalWrite(a , LOW);
    digitalWrite(b , LOW);
    digitalWrite(d , LOW);
    digitalWrite(e , LOW);
    digitalWrite(g , LOW);
    digitalWrite(dp , LOW);
  }
  if (val == 3){
    digitalWrite(a , HIGH);
    digitalWrite(b , HIGH);
    digitalWrite(g , HIGH);
    digitalWrite(c , HIGH);
    digitalWrite(d , HIGH);
    digitalWrite(dp , HIGH);
    delay(1000);
    digitalWrite(a , LOW);
    digitalWrite(b , LOW);
    digitalWrite(g , LOW);
    digitalWrite(c , LOW);
```

```
    digitalWrite(d , LOW);
    digitalWrite(dp , LOW);
  }
  if (val == 4){
    digitalWrite(b , HIGH);
    digitalWrite(c , HIGH);
    digitalWrite(f , HIGH);
    digitalWrite(g , HIGH);
    digitalWrite(dp , HIGH);
    delay(1000);
    digitalWrite(b , LOW);
    digitalWrite(c , LOW);
    digitalWrite(f , LOW);
    digitalWrite(g ,LOW);
    digitalWrite(dp , LOW);
  }
  if (val == 5) {
    digitalWrite(a , HIGH);
    digitalWrite(c , HIGH);
    digitalWrite(d , HIGH);
    digitalWrite(f , HIGH);
    digitalWrite(g , HIGH);
    digitalWrite(dp , HIGH);
    delay(1000);
    digitalWrite(a , LOW);
    digitalWrite(c , LOW);
    digitalWrite(d , LOW);
    digitalWrite(f , LOW);
    digitalWrite(g ,LOW);
    digitalWrite(dp , LOW);
  }
  if (val == 6) {
    digitalWrite(a , HIGH);
    digitalWrite(c , HIGH);
```

```
    digitalWrite(d , HIGH);
    digitalWrite(e , HIGH);
    digitalWrite(f , HIGH);
    digitalWrite(g , HIGH);
    digitalWrite(dp , HIGH);
    delay(1000);
    digitalWrite(a , LOW);
    digitalWrite(c , LOW);
    digitalWrite(d , LOW);
    digitalWrite(e , LOW);
    digitalWrite(f , LOW);
    digitalWrite(g ,LOW);
    digitalWrite(dp , LOW);
  }
  if (val == 7){
    digitalWrite(a , HIGH);
    digitalWrite(b , HIGH);
    digitalWrite(c , HIGH);
    digitalWrite(dp , HIGH);
    delay(1000);
    digitalWrite(a , LOW);
    digitalWrite(b , LOW);
    digitalWrite(c , LOW);
    digitalWrite(dp , LOW);
  }
  if (val == 8){
    digitalWrite(a , HIGH);
    digitalWrite(b , HIGH);
    digitalWrite(c , HIGH);
    digitalWrite(d , HIGH);
    digitalWrite(e , HIGH);
    digitalWrite(f , HIGH);
    digitalWrite(g , HIGH);
    digitalWrite(dp , HIGH);
```

```
    delay(1000);
    digitalWrite(a , LOW);
    digitalWrite(b , LOW);
    digitalWrite(c , LOW);
    digitalWrite(d , LOW);
    digitalWrite(e , LOW);
    digitalWrite(f , LOW);
    digitalWrite(g ,LOW);
    digitalWrite(dp , LOW);
  }
  if (val == 9){
    digitalWrite(a , HIGH);
    digitalWrite(b , HIGH);
    digitalWrite(c , HIGH);
    digitalWrite(d , HIGH);
    digitalWrite(f , HIGH);
    digitalWrite(g , HIGH);
    digitalWrite(dp , HIGH);
    delay(1000);
    digitalWrite(a , LOW);
    digitalWrite(b , LOW);
    digitalWrite(c , LOW);
    digitalWrite(d , LOW);
    digitalWrite(f , LOW);
    digitalWrite(g ,LOW);
    digitalWrite(dp , LOW);
  }
}
```

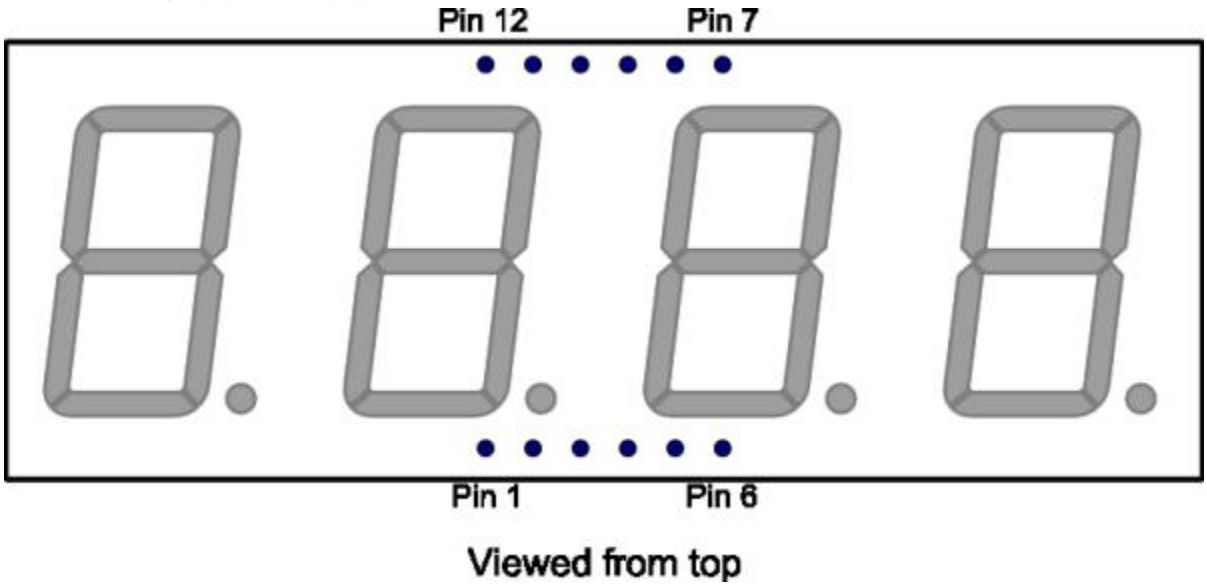
33> 4 DIGITAL TUBE X 1

4-Digit 7-Segment 입니다.

4 개의 숫자 & DOT 표시 가능한 모듈입니다.

캐소드 타입 4 자리 LED 모듈입니다.

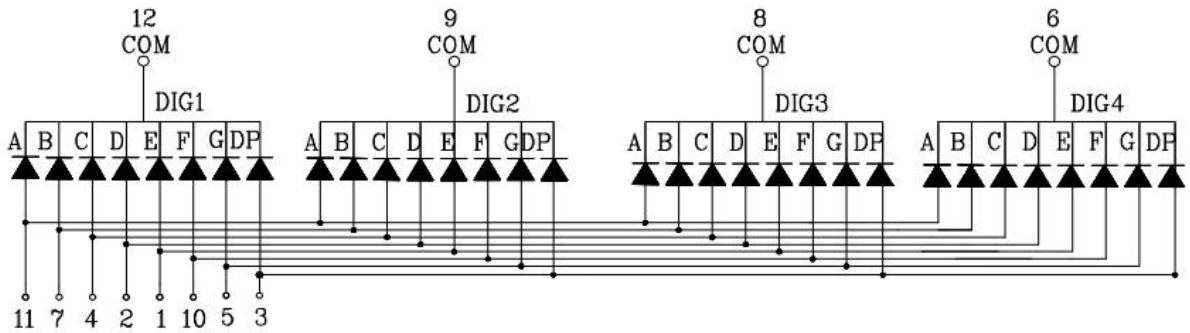
핀 번호는 아래와 같습니다.



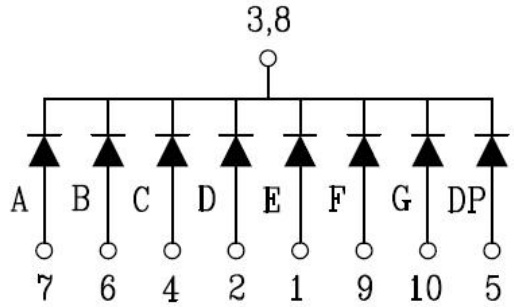
4-Digit 모듈 모델 넘버 인쇄된 부분 (SH8461AS 찍혀 있는 부분) 아래쪽으로 향하게 하고 핀 번호는 위의 이미지를 참조합니다.

데이터 쉬트는 아래의 링크를 참조합니다.(기술지원 게시판에도 있습니다)
http://www.igameplus.com/pds/gpshop/arduino/shield_module/4digit-7segment/4digit-7segment-datasheet.pdf

데이터 쉬트 내용 중 Circuit Diagram 을 확인합니다.



1-Digit 7-Segment 모듈이 4 개 12,9,8,6 핀이 Common 핀이면서 그라운드 표시입니다.
캐소드 방식의 7-Segment 입니다. 그라운드 표시가 12,9,8,6 번입니다.
참고사항으로 아래의 이미지는 위에서 0 보았던 1-Digit 7-Segment 의 회로도 입니다.



4-Digit 표시 모듈은 1-Digit 4 개를 병렬 연결한 것을 알게 됩니다.
그리고 12,9,8,6 번 핀은 1-Digit 의 3,8 번 핀의 역할을 하게 됩니다.

이제 아두이노 보드와 와이어링을 해봅시다.
4 Digit 7 Segment 1 번부터 12 번까지의 핀이 있습니다.

각각의 핀 기능은 아래의 기능 아두이노와의 연결은 위의 이미지를 참조하여 아래의 표와 같이 합니다.

모듈 핀	아두이노	Segment
1	D10	segE
2	D9	segD
3	D13	segDP
4	D8	segC
5	D12	segG
6	D5	
7	D7	segB
8	D4	
9	D3	
10	D11	segF
11	D6	segA
12	D2	

아두이노 공식 사이트의 링크 SevSeg 라이브러리를 사용 하도록 합니다.
7-Segment 를 편리하게 사용하도록 지원되는 클래스 라이브러리입니다.

http://playground.arduino.cc/Main/SevenSegmentLibrary

라이브러리 적용 후 스케치 코드에서는
#include <SevSeg.h> 선언하고 사용 하게 됩니다.

아두이노&스케치 프로그램의 장점은 기 구현된 많은 라이브러리들이 존재 한다는 것입니다. 최소한의 C/C++ 언어만 알아도 많은 것들을 활용 해볼 수도 있습니다.

스케치 예제 코드입니다.

```
#include "SevSeg.h"

//Create an instance of the object.
// 글로벌 클래스 변수입니다. (전역 클래스 변수)
SevSeg myDisplay;

//Create global variables
unsigned long timer;
int deciSecond = 0;

void setup()
{

    int displayType = COMMON_CATHODE; //캐소드 모듈 지정.


     //This pinout is for a regular display
     int digit1 = 2; //Pin 12 on my 4 digit display
     int digit2 = 3; //Pin 9 on my 4 digit display
     int digit3 = 4; //Pin 8 on my 4 digit display
     int digit4 = 5; //Pin 6 on my 4 digit display

     //Declare what pins are connected to the segments
     int segA = 6; //Pin 11 on my 4 digit display
     int segB = 7; //Pin 7 on my 4 digit display
```

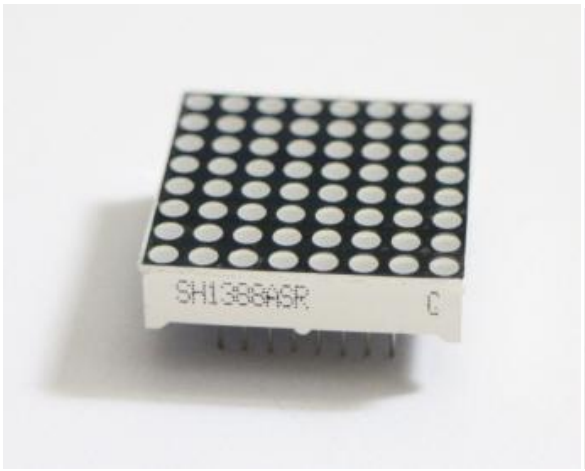
```
	int segC = 8; //Pin 4 on my 4 digit display
	int segD = 9; //Pin 2 on my 4 digit display
	int segE = 10; //Pin 1 on my 4 digit display
	int segF = 11; //Pin 10 on my 4 digit display
	int segG = 12; //Pin 5 on my 4 digit display
	int segDP= 13; //Pin 3 on my 4 digit display

	int numberOfDigits = 4;   //Do you have a 1, 2 or 4 digit display?

	myDisplay.Begin(displayType, numberOfDigits, digit1, digit2, digit3, digit4, segA, segB, segC, segD, segE, segF, segG, segDP);

	myDisplay.SetBrightness(100); //Set the display to 100% brightness level

	timer = millis();
}

void loop()
{
	//Example ways of displaying a decimal number
	char tempString[10]; //Used for sprintf
	sprintf(tempString, "%04d", deciSecond); //Convert deciSecond into a string that is right adjusted
	//sprintf(tempString, "%d", deciSecond); //Convert deciSecond into a string that is left adjusted
	//sprintf(tempString, "%04d", deciSecond); //Convert deciSecond into a string with leading zeros
	//sprintf(tempString, "%4d", deciSecond * -1); //Shows a negative sign infront of right adjusted number
	//sprintf(tempString, "%4X", deciSecond); //Count in HEX, right adjusted

	//Produce an output on the display
	myDisplay.DisplayString(tempString, 8); //(numberToDisplay, decimal point location)
```

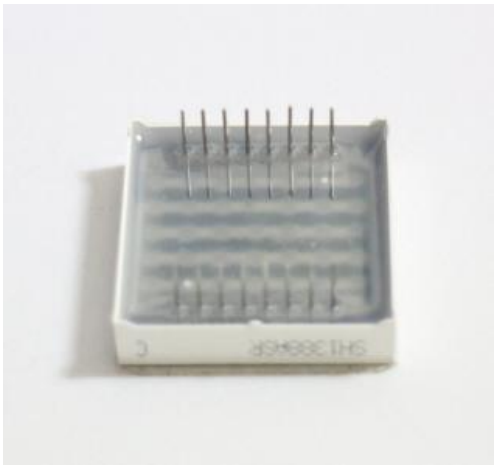
```
    //Other examples
    //myDisplay.DisplayString(tempString, 0); //Display string, no decimal point
    //myDisplay.DisplayString("-23b", 3); //Display string, decimal point in third
position

    //Check if 100 ms has elapsed
    // 100 ms, 0.1 초
    if (millis() - timer >= 100)
    {
        timer = millis();
        deciSecond++;
    }

    delay(5);
}
```

와이어링 및 업로드 정상 상태인 경우 숫자 증가 하는 것을 볼 수 있습니다.
위의 코드중 myDisplay.DisplayString(tempString, 8);
DisplayString 이라는 함수가 있습니다.
SevSeg.h 파일을 열어보면
void SevSeg::DisplayString(char* toDisplay, byte DecAposColon); 라고 정의 되어 있습니다. 2 번째 파라메터는 DOT 표시입니다.
myDisplay.DisplayString(tempString, 0); // DOT 표시 안함.
myDisplay.DisplayString(tempString, 1); // DOT 표시 1 번째 위치.
myDisplay.DisplayString(tempString, 2); // DOT 표시 2 번째 위치.
myDisplay.DisplayString(tempString, 4); // DOT 표시 3 번째 위치.
myDisplay.DisplayString(tempString, 8); // DOT 표시 4 번째 위치.
2 번째와 3 번째 위치에 DOT 을 표시하고 싶다면
myDisplay.DisplayString(tempString, 2 | 4); // OR 연산자 or 6

34> 8 * 8 DOT MATRIX MODULE X 1

8 x 8 사이즈의 DOT MATRIX 모듈입니다.



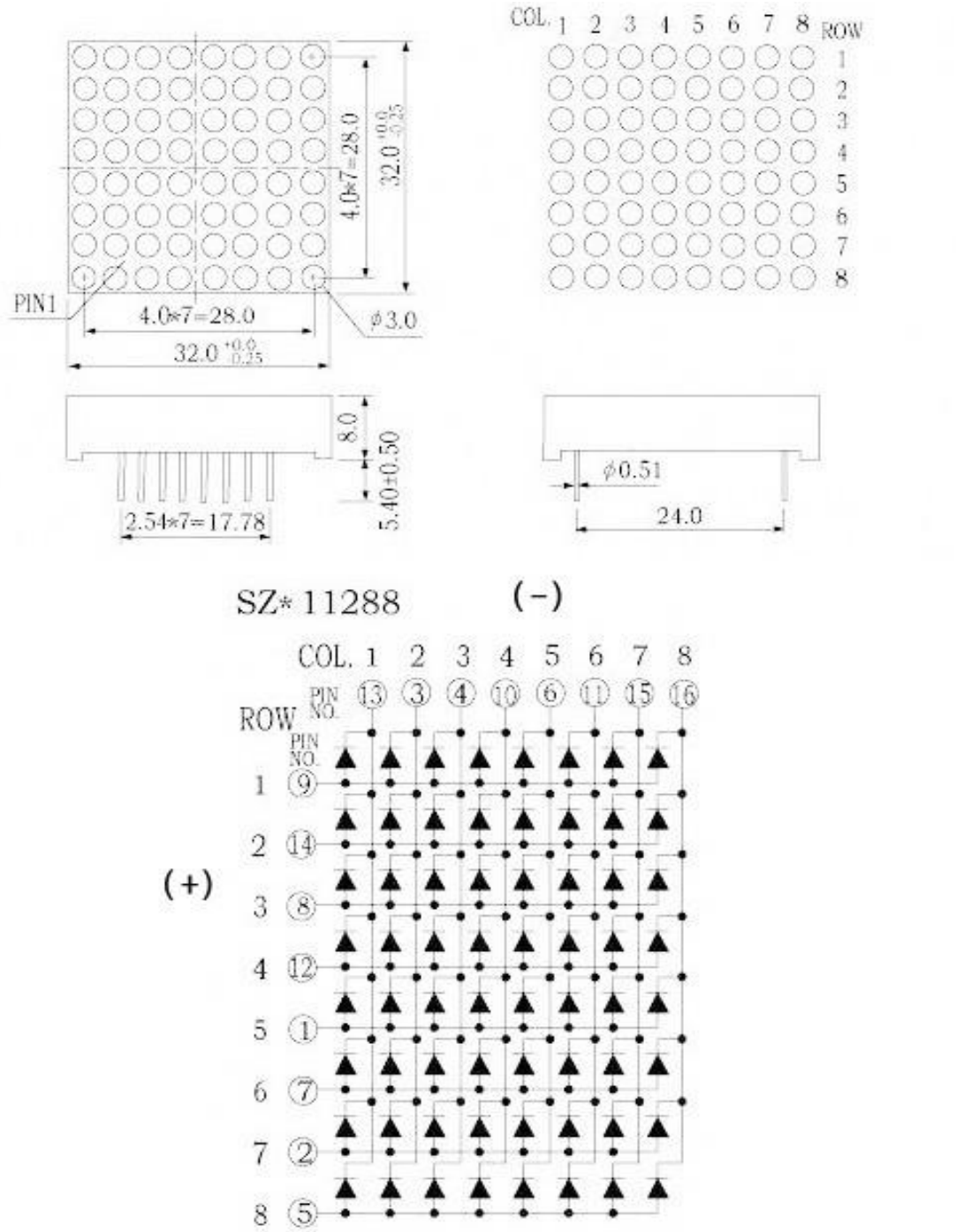


가로 8 칸 세로 8 칸으로 LED 가 교차 연결된 모듈 입니다.

Dot: 3mm
색상: RED
크기: 32mm x 32 mm x 8 mm(height)

8x8 pixel 로 표현 가능한 모든 것을 디스플레이 할 수 있습니다.
64 개의 교차 방식의 LED 로 구성된 모듈입니다.
고급 디지털시계, 시스템 상황 표시(선명한), 경고 표시 등등 여러 가지 용도로 사용되고 있습니다. 실제 설치 및 제품화될 때의 LED 모듈은 LED 사이즈 대, 중, 소 등의 크기가 용도별로 다를 뿐 구현 내용이나 모듈 연결 방식은 비슷합니다.

아래의 참조 회로도를 봅니다.



전체 크기와 핀 간격, 그리고 핀 번호, 직병렬 회로도가 있습니다. 세모꼴 모양은 LED 다이오드 표시입니다.

"LED", 정확한 의미는 "Light Emitting Diode"

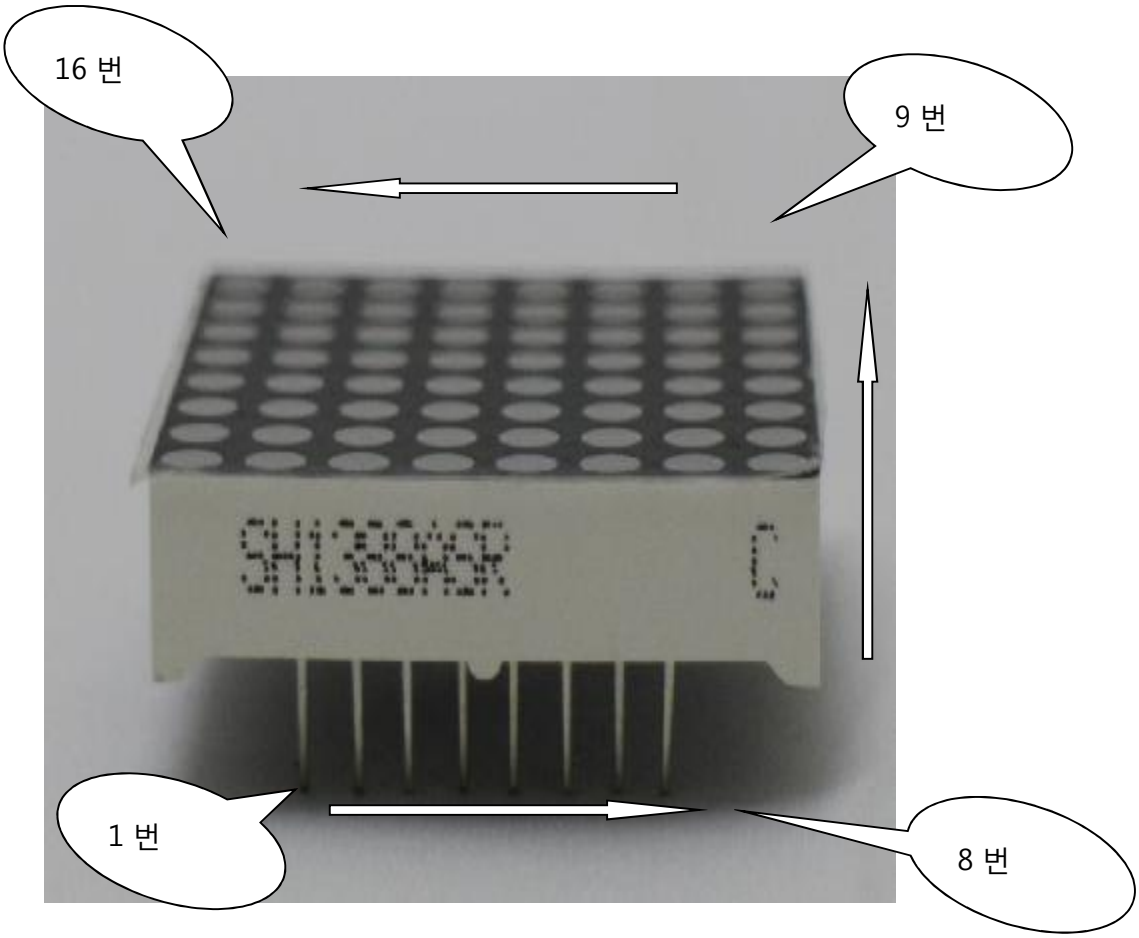
즉, 빛 발산 2 극관, 발광 다이오드입니다. 즉 (+), (-)소자입니다. 회로도의 세모꼴 모양을 보시면 (+)와 (-) 정확히 구분된 것을 보실 수 있습니다.

LED 는 아시다시피 (+),(-) 연결 해야 켜집니다.
아래의 회로도를 보면 ROW 는 (+), COL 은 (-) 되어 있습니다.
COL 3 번째, ROW 5 번째의 LED 를 켜고 싶다면, ROW 5 번째 (+)를, COL 3 번째를 (-) 연결 해주면 됩니다. 아두이노 보드에 연결 되었다면 ROW 5 번째 핀을 HIGH, COL 3 번째 핀을 LOW 로 하면 됩니다.

이제 아두이노 우노 R3 에서 컨트롤 하기 위해서 16 핀 모두 사용 합니다.
그런데, 아두이노 우노 R3 는 디지털핀은 0~13, 14 개 입니다. 2 개 모자랍니다.
결국 아날로그 핀도 사용합니다.
대부분 사용되는 핀수를 줄이기 위해서 74HC595 종류의 칩을 사용 합니다.
74HC595 칩은 3 개의 핀으로 8 개의 OUTPUT 신호를 컨트롤 합니다.
그래도 6 핀을 사용하게 됩니다. 결국, 2 개의 74HC595 필요하게 됩니다.

본 예제에서는 다이렉트로 모두 연결 해보도록 합니다. 아날로그 핀도 디지털 핀처럼 사용할 수 있습니다.

아래의 그림처럼 보통 핀 번호는 마크,모델 명칭이 있는 부분의 맨 왼쪽부터 1 번째 번호입니다.
1 번부터 끝 번호까지는 반시계 방향(CCW) 으로 순차적으로 부여 해서 사용합니다.
아래의 이미지처럼 넘버링 됩니다.

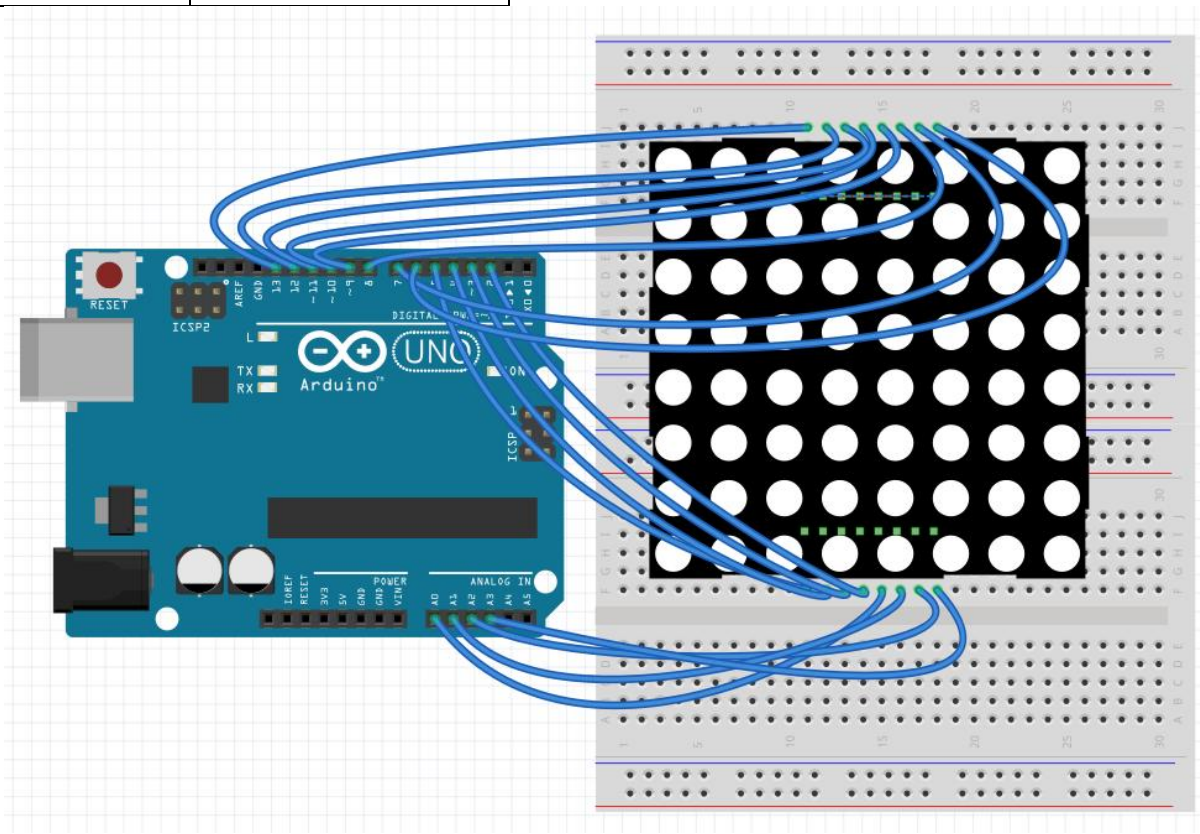


아두이노 보드와의 와이어링은 아래의 표와 그림을 참조합니다.

5, 4, 3, 2, 14, 15, 16, 17, 13, 12, 11, 10, 9, 8, 7, 6

모듈 핀 번호	아두이노
1	5
2	4
3	3
4	2
5	A0 (14)
6	A1 (15)
7	A2 (16)
8	A3 (17)
9	13
10	12
11	11
12	10
13	9

14	8
15	7
16	6



와이어링 예시 이미지입니다

200~220 Ohm 저항을 사용해야 합니다. 사용하는 이유는 아두이노에서 HIGH 신호 전류가 5V 가량 나옵니다. LED 는 거의 대부분 2V~3V 동작하기 때문에 높은 전압으로 사용하게 되면 오 동작 및 에러가 발생될 수 있습니다. 정확하고 안전한 동작을 위해서는 전압을 조절 해 주기 위해 저항을 사용합니다. 그럼 220 R Ohm 또는 300 Ohm 저항을 사용하는지 알아 봅니다.
저항 계산식은 아래와 같습니다.

기호 범례: V: 전압 I: 전류(A) R: 저항(옴) P: 전력(w)

P=VI (전력=전압 x 전류)
V=IR (전압=전류 x 저항)
I=V/R (전류=전압/저항)

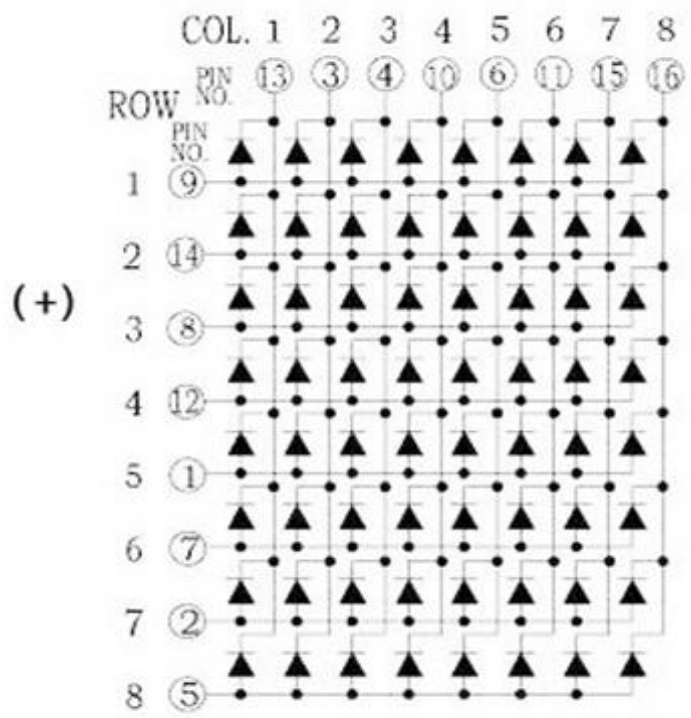
R=V/I (저항=전압/전류)
아두이노 보드의 핀 HIGH 신호는 5V 출력 됩니다.
5V 전압에 2V 10mA LED 를 사용 한다면
5V-2V = 3V 즉, 5V 전력에 LED 에 적용되는 전력은 2V 이고 나머지 3V 저항에서 처리해 주도록 합니다.
LED 모듈은 2V 10mA 동작 전원이라면
R=V/I → 3V / 0.01A = 200 Ohm 계산됩니다.

8x8 DOT 매트릭스 예제 코드입니다.
KIT 예제 중 가장 어려운 HELLO 예제 이기도 합니다.
HELLO 라는 각각의 구성 문자를 사이드 스크롤 하는 예제입니다.
보통 옥외 LED 간판에 많이 사용되는(오른쪽->왼쪽) 스크롤 방식입니다.

```
#include <FrequencyTimer2.h>

#define SPACE { ₩
      {0, 0, 0, 0, 0, 0, 0, 0},   ₩
      {0, 0, 0, 0, 0, 0, 0, 0}, ₩
      {0, 0, 0, 0, 0, 0, 0, 0}, ₩
      {0, 0, 0, 0, 0, 0, 0, 0}, ₩
      {0, 0, 0, 0, 0, 0, 0, 0}, ₩
      {0, 0, 0, 0, 0, 0, 0, 0}, ₩
      {0, 0, 0, 0, 0, 0, 0, 0}, ₩
      {0, 0, 0, 0, 0, 0, 0, 0} ₩
}

#define H { ₩
      {0, 1, 0, 0, 0, 0, 1, 0}, ₩
      {0, 1, 0, 0, 0, 0, 1, 0}, ₩
      {0, 1, 0, 0, 0, 0, 1, 0}, ₩
      {0, 1, 1, 1, 1, 1, 1, 0}, ₩
      {0, 1, 0, 0, 0, 0, 1, 0}, ₩
```

```
        {0, 1, 0, 0, 0, 0, 1, 0}, ₩
        {0, 1, 0, 0, 0, 0, 1, 0}, ₩
        {0, 1, 0, 0, 0, 0, 1, 0}   ₩
}

#define E   { ₩
        {0, 1, 1, 1, 1, 1, 1, 0}, ₩
        {0, 1, 0, 0, 0, 0, 0, 0}, ₩
        {0, 1, 0, 0, 0, 0, 0, 0}, ₩
        {0, 1, 1, 1, 1, 1, 1, 0}, ₩
        {0, 1, 0, 0, 0, 0, 0, 0}, ₩
        {0, 1, 0, 0, 0, 0, 0, 0}, ₩
        {0, 1, 0, 0, 0, 0, 0, 0}, ₩
        {0, 1, 1, 1, 1, 1, 1, 0}   ₩
}

#define L { ₩
        {0, 1, 0, 0, 0, 0, 0, 0}, ₩
        {0, 1, 0, 0, 0, 0, 0, 0}, ₩
        {0, 1, 0, 0, 0, 0, 0, 0}, ₩
        {0, 1, 0, 0, 0, 0, 0, 0}, ₩
        {0, 1, 0, 0, 0, 0, 0, 0}, ₩
        {0, 1, 0, 0, 0, 0, 0, 0}, ₩
        {0, 1, 0, 0, 0, 0, 0, 0}, ₩
        {0, 1, 1, 1, 1, 1, 1, 0}   ₩
}

#define O { ₩
        {0, 0, 0, 1, 1, 0, 0, 0}, ₩
        {0, 0, 1, 0, 0, 1, 0, 0}, ₩
        {0, 1, 0, 0, 0, 0, 1, 0}, ₩
        {0, 1, 0, 0, 0, 0, 1, 0}, ₩
        {0, 1, 0, 0, 0, 0, 1, 0}, ₩
        {0, 1, 0, 0, 0, 0, 1, 0}, ₩
```

```
    {0, 0, 1, 0, 0, 1, 0, 0}, ₩
    {0, 0, 0, 1, 1, 0, 0, 0}   ₩
}

byte col = 0;
byte leds[8][8];

// pin[xx] on led matrix connected to nn on Arduino (-1 is dummy to make array start at pos 1)
int pins[17]= {-1, 5, 4, 3, 2, 14, 15, 16, 17, 13, 12, 11, 10, 9,8, 7, 6};

// 아래의 코드는 위에서 보았던 내부 핀 순서를 참조합니다.
```

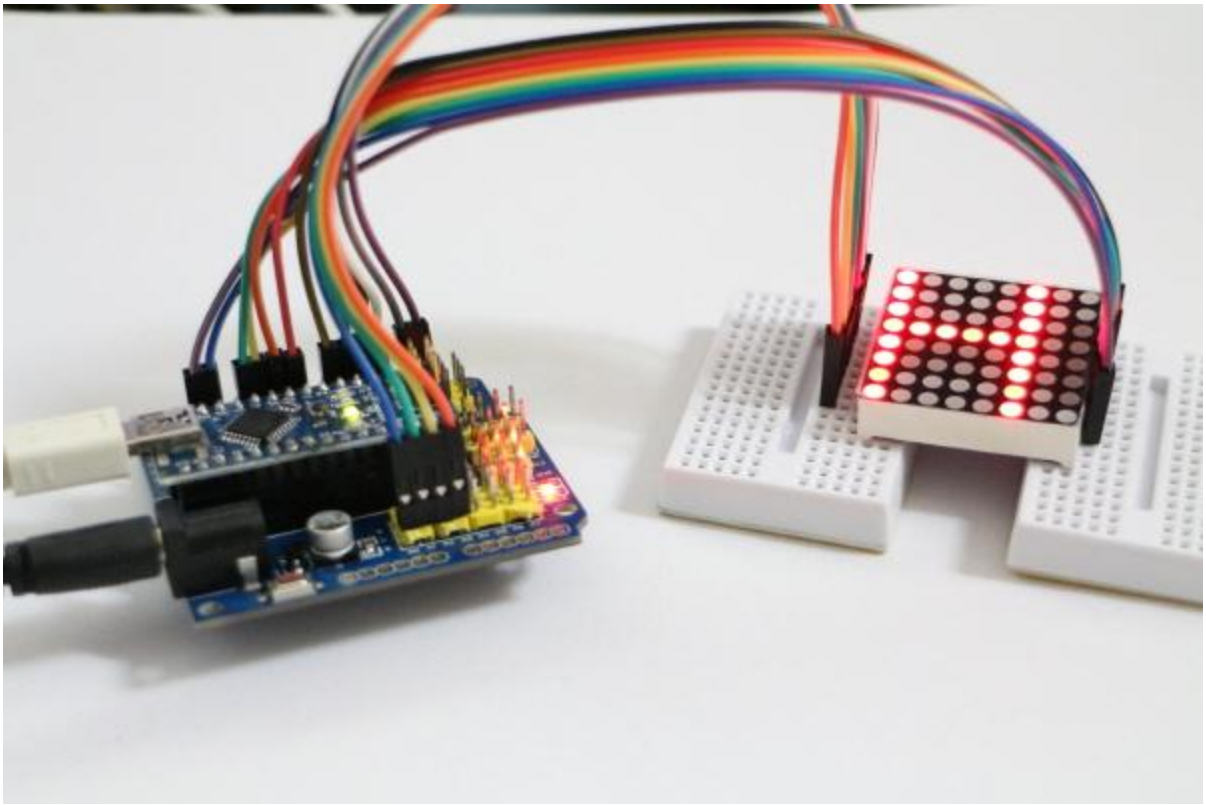


```
// col[xx] of leds = pin yy on led matrix
// 행 8 개 순서대로 지정합니다.
//
int cols[8] = {pins[13], pins[3], pins[4], pins[10], pins[06],pins[11], pins[15], pins[16]};

// row[xx] of leds = pin yy on led matrix
// 열 8 개 순서대로 지정해줍니다.
int rows[8] = {pins[9], pins[14], pins[8], pins[12], pins[1],pins[7], pins[2], pins[5]};

const int numPatterns = 6;
byte patterns[numPatterns][8][8] = {
  H,E,L,L,O,SPACE
};

int pattern = 0;

void setup() {
  // sets the pins as output
  for (int i = 1; i <= 16; i++) {
    pinMode(pins[i], OUTPUT);
  }

  // set up cols and rows
  for (int i = 1; i <= 8; i++) {
    digitalWrite(cols[i - 1], !LOW);
  }

  for (int i = 1; i <= 8; i++) {
    digitalWrite(rows[i - 1], !LOW);
  }

  clearLeds();

  // Turn off toggling of pin 11
```

```
  FrequencyTimer2::disable();
  // Set refresh rate (interrupt timeout period)
  FrequencyTimer2::setPeriod(2000);
  // Set interrupt routine to be called
  FrequencyTimer2::setOnOverflow(display);

  setPattern(pattern);
}

void loop() {
    pattern = ++pattern % numPatterns;
    slidePattern(pattern, 100);
}

void clearLeds() {
  // Clear display array
  for (int i = 0; i < 8; i++) {
    for (int j = 0; j < 8; j++) {
      leds[i][j] = 0;
    }
  }
}

void setPattern(int pattern) {
  for (int i = 0; i < 8; i++) {
    for (int j = 0; j < 8; j++) {
      leds[i][j] = patterns[pattern][i][j];
    }
  }
}

void slidePattern(int pattern, int del) {
  for (int l = 0; l < 8; l++) {
    for (int i = 0; i < 7; i++) {
```

```
        for (int j = 0; j < 8; j++) {
          leds[j][i] = leds[j][i+1];
        }
      }
      for (int j = 0; j < 8; j++) {
        leds[j][7] = patterns[pattern][j][0 + l];
      }
      delay(del);
    }
}


// Interrupt routine
void display() {
  digitalWrite(cols[col], !LOW);   // Turn whole previous column off
  col++;
  if (col == 8) {
    col = 0;
  }
  for (int row = 0; row < 8; row++) {
    if (leds[col][7 - row] == 1) {
      digitalWrite(rows[row], !LOW);   // Turn on this led
    }
    else {
      digitalWrite(rows[row], !HIGH); // Turn off this led
    }
  }
  digitalWrite(cols[col], !HIGH); // Turn whole column on at once (for equal lighting
times)
}
```

위의 코드와 이미지는 아두이노 나노(확장보드)에서 테스트된 코드입니다.
물론 아두이노 우노에서의 와이어링은 동일합니다



.

제대로 와이어링 & 코드를 적용한 경우에는 아래와 같은 동영상 처럼 실행됩니다.

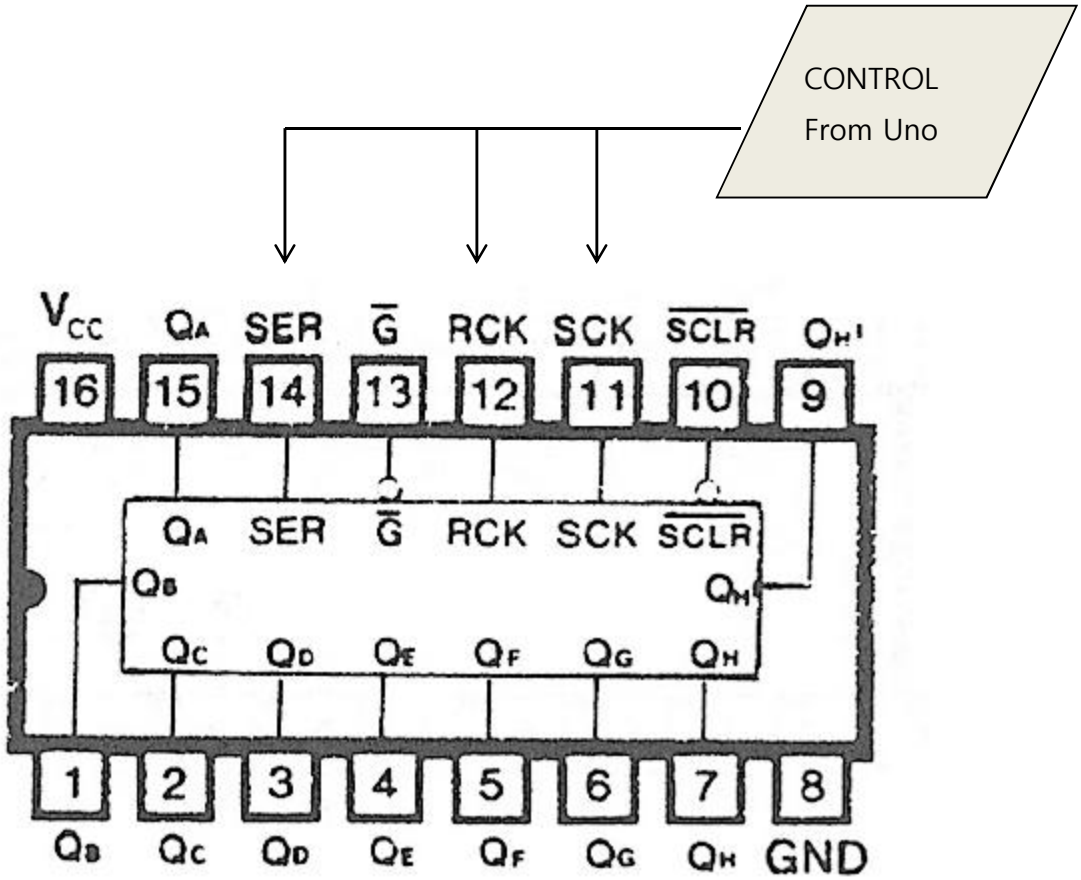
Right To Left “HELLO” 스크롤입니다.
(기술지원 게시판에도 바로가기 링크 있습니다)
http://www.youtube.com/watch?v=8nmkUlYU1PA

35> 74HC595N CHIPS X 1

8 비트 쉬프트 레지스터 IC 칩입니다.
8-Bit Serial-Parallel Shift Register 3-State

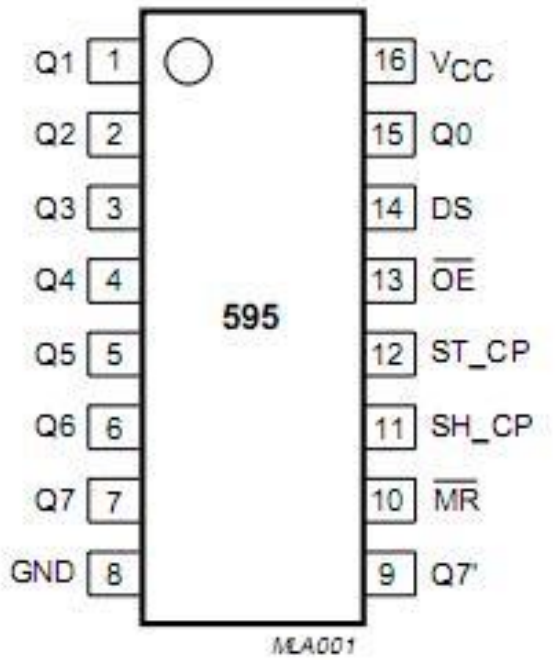
여러 개의 OUTPUT 방향 신호 처리할 때 많이 사용되는 칩입니다.
시리얼로 데이터를 받아서 병렬로 출력 해주는 칩입니다.
Only Direction Output 신호 아웃풋 전용으로만 사용되는 특성을 가진 칩입니다.
3 개의 핀을 사용하여 8 개의 출력 전용(OUTPUT Only) 컨트롤을 가능하게 합니다.
아두이노 연결에는 디지털 핀에 연결하면 됩니다.

지금 사용할 8 비트 쉬프트 칩셋 가용 전원은 대부분 2V ~ 6V 이므로 아두이노에서 쉽게 사용 가능합니다



QA ~ QH → 74HC595N 출력 포트입니다.

일반적으로 적용 가능한 용도는 LED, FND, DOT MATRIX LED 등에 많이 사용 됩니다.



쉬프트 레지스터 사용시, 아두이노 스케치 기본 라이브러리에서 bitWrite, bitRead, shiftOut,shiftIn 함수가 지원됩니다.

```
#define bitRead(value, bit) (((value) >> (bit)) & 0x01)
#define bitSet(value, bit) ((value) |= (1UL << (bit)))
#define bitClear(value, bit) ((value) &= ~(1UL << (bit)))
#define bitWrite(value, bit, bitvalue) (bitvalue ? bitSet(value, bit) : bitClear(value, bit))
```

위의 정의(define)된 선언은 Defined from #include <Arduino.h>

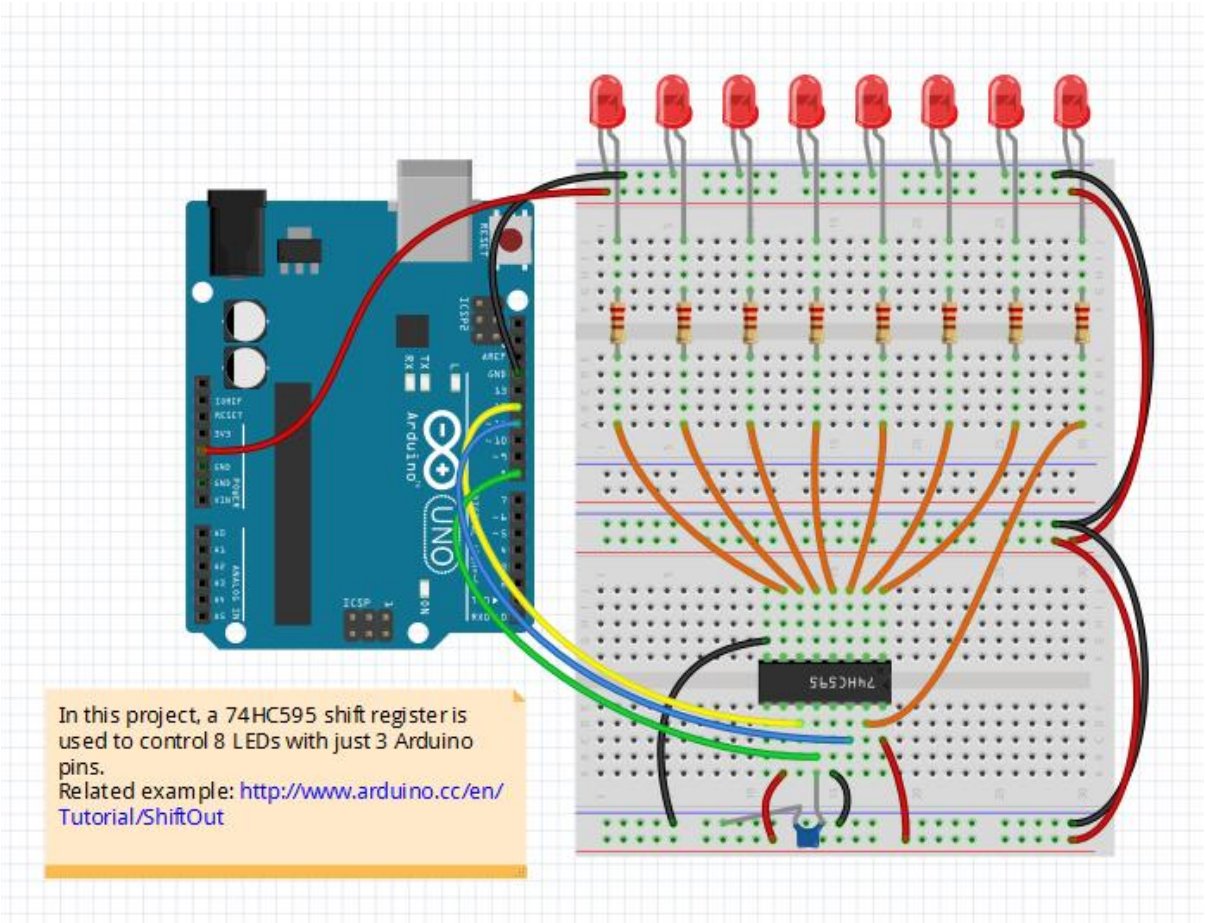
위의 비트 관련 함수는 비트 연산 함수(매크로 정의)로서, 하드웨어 개발, 소프트웨어 개발에도 많이 사용 됩니다. 가능하면 이해를 하시는게 좋습니다.

Shift Out 에 관한 내용은 아두이노 공식 사이트에도 상세히 설명 되어 있습니다.
http://www.arduino.cc/en/Tutorial/ShiftOut

위 링크 주소에 설명 되어 있는 내용들은 가능하면 이해하고 넘어가시면 차후에 여러가지 용도에 맞는 하드웨어 개발 및 응용 개발에 많은 도움이 되시리라 봅니다. 특히 여러 개의 745HC595 사용에 대한 설명도 주의 깊게 살펴 보시기 바랍니다. 그리고 74HC595 와 비슷한 칩으로는 74LS164,74C299 등도 있습니다.

74HC595 칩을 사용 해봅시다. 8 개의 OUTPUT 신호 HIGH/LOW 기능이 되므로 8 개의 LED 컨트롤 해보도록 합니다.
아두이노 사이트에서 친절하게 설명 & 예제 있습니다.
http://www.arduino.cc/en/Tutorial/ShiftOut

사용되는 부품은 74HC595N x 1 개, 220 Ohm R x 8 개, 100nF 커패시터 1 개, 점퍼 선 필요합니다.



위의 코드는 아래와 같습니다.
시리얼 인풋으로 '0' ~'9' 입력을 받아 숫자로 변환합니다. ASCII 코드 '0' ~ '9' 를 정수로 변환하는 방법은 많이 있지만, 아래의 예제 코드는 입력 받은 값에 마이너스 48 을 해주고 있습니다.

```
int bitToSet = Serial.read() - 48;
```

Ascii 코드 테이블에 대한 내용은 http://en.wikipedia.org/wiki/ASCII 에도 상세히 설명 되어 있습니다. 해당 되는 '0' ~ '9' 아스키 코드값을 찾아 보기 바랍니다.

대입된 숫자 인덱스에 의해 LED 를 ON/OFF 예제입니다.

```
/*
  Shift Register Example
 for 74HC595 shift register

 This sketch turns reads serial input and uses it to set the pins
 of a 74HC595 shift register.

 Hardware:
 * 74HC595 shift register attached to pins 2, 3, and 4 of the Arduino,
 as detailed below.
 * LEDs attached to each of the outputs of the shift register

 Created 22 May 2009
 Created 23 Mar 2010
 by Tom Igoe

 */

//Pin connected to latch pin (ST_CP) of 74HC595
const int latchPin = 8;
```

```
//Pin connected to clock pin (SH_CP) of 74HC595
const int clockPin = 12;
////Pin connected to Data in (DS) of 74HC595
const int dataPin = 11;

void setup() {
      //set pins to output because they are addressed in the main loop
      pinMode(latchPin, OUTPUT);
      pinMode(dataPin, OUTPUT);
      pinMode(clockPin, OUTPUT);

      Serial.begin(9600);
      Serial.println("reset");
}

void loop()
{
      if (Serial.available() > 0)
      {
            // ASCII '0' through '9' characters are
            // represented by the values 48 through 57.
            // so if the user types a number from 0 through 9 in ASCII,
            // you can subtract 48 to get the actual value:
            int bitToSet = Serial.read() - 48;

            // write to the shift register with the correct bit set high:
            registerWrite(bitToSet, HIGH);
      }
}

// This method sends bits to the shift register:
void registerWrite(int whichPin, int whichState)
{
      // the bits you want to send
```

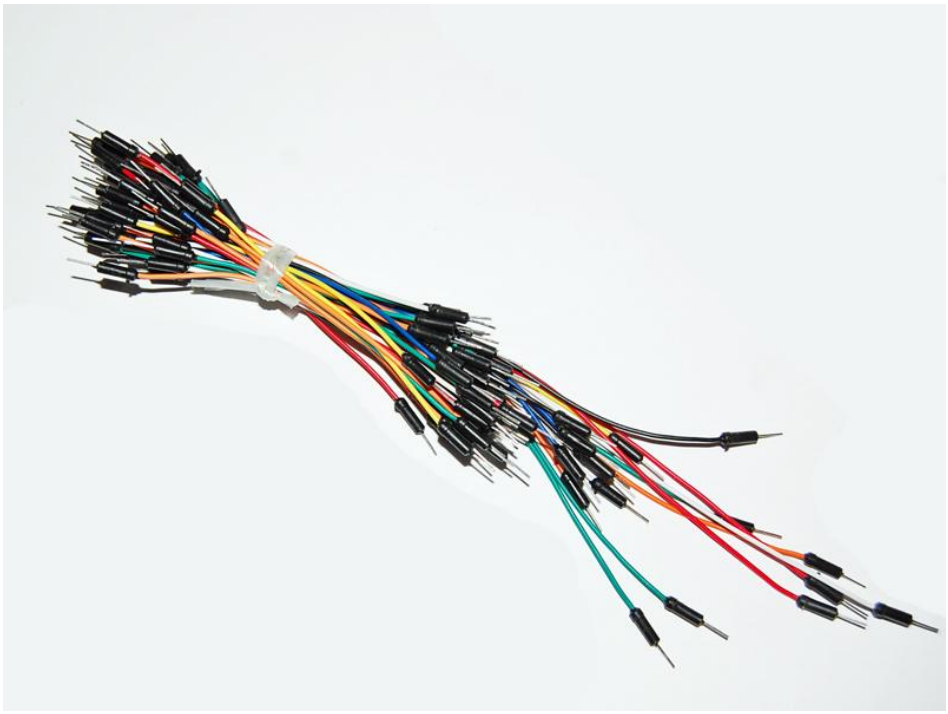
```
    byte bitsToSend = 0;

    // turn off the output so the pins don't light up
    // while you're shifting bits:
    digitalWrite(latchPin, LOW);

    // turn on the next highest bit in bitsToSend:
    bitWrite(bitsToSend, whichPin, whichState);

    // shift the bits out:
    shiftOut(dataPin, clockPin, MSBFIRST, bitsToSend);

    // turn on the output so the LEDs can light up:
    digitalWrite(latchPin, HIGH);
}
```

36> INFRARED REMOTE CONTROL X 1

IR 리모트 컨트롤러. 적외선 리모트 컨트롤러입니다.

37> BREADBOARD JUMPER X 65

브레드보드 점퍼 와이어 65 개 (입고 순서 상황에 따라 75 개일수도 있습니다)



아두이노 보드와 각종 브레드보드 연결 시에 필요한 점퍼 케이블입니다.

38> FEMALE TO MALE DUPONT LINES X 10

30cm 30pcs Female-Male Dupont Breadboard Jumper Cable Wire Line 2.54mm pitch.

뒤퐁 (듀폰) 케이블입니다. 한쪽은 헤더핀에 연결 할 수 있고 한쪽은 아두이노 보드 핀에 꼽아서 사용 할 수 있습니다.
2.54 피치 간격이 유지되는 케이블입니다. 즉, 아두이노 보드의 Female 헤더 홀더 간격과 정확히 일치되는 케이블입니다. 동시에 여러 개를 이어서 홀딩 할 경우에도 밀리지 않고 정확히 연결 할 수 있습니다.

39> 9 VOLT BATTERY SNAP X 1

9 볼트 전용 배터리를 사용 할 경우 필요합니다.

대부분 9V 전용 배터리는 스넵 케이블에 연결하여 아두이노 우노 R3 보드의 전원 공급으로 사용됩니다.

40> USB CABLE X 1

B 타입 USB 케이블입니다.
아두이노 보드와 PC USB 포트와 연결 후 스케치 프로그램 업로드 시에 사용되는 케이블입니다.

아두이노는 프로그래밍 & 디버깅을 위해 가상 시리얼 포트를 할당하여 사용됩니다.

아두이노 우노 R3 보드는 USB 시리얼 통신을 하기 위해 ATMEGA16U2 IC 있습니다.

내부적으로 ATMEGA16U2 는 ATMEGA328P-PU 와 SPI 방식으로 연결되어 있습니다.

41> HC-SR04 x 1 초음파 센서

초음파 거리 측정 센서 입니다.

4 핀으로 구성된 모듈입니다. 초음파를 발사하여 되돌아 오는 경과 시간을 기준으로 거리를 측정할 수 있습니다.
유효 거리는 2 cm ~ 400 cm 입니다. 500 cm 도 가능하지만 안정적인 유효 측정 저리는 400 cm 로 계산하면 됩니다.

데이터 쉬트 http://www.igameplus.com/pds/gpshop/arduino/pdf/HCSR04.pdf

와이어링은 아래의 표를 참조합니다.

센서 핀	아두이노 핀
VCC	5V
TRIG	D12
ECHO	D13
GND	GND

TRIG 핀 연결은 디지털 신호입니다.
ECHO 핀 연결은 아날로그 입력입니다.

41.1 DIRECT TRIG, ECHO 사용

스케치 코드입니다.

```
#define trigPin 12
#define echoPin 13

void setup() {

  // 시리얼 포트 9600 속도 초기화.
  Serial.begin (9600);
  // 센서 Trig 연결 포트를 출력으로 설정합니다.
  pinMode (trigPin, OUTPUT);
  // 센서 에코 연결핀은 입력 모드로 설정.
  pinMode (echoPin, INPUT);
}

void loop() {
  // 지속시간 , 거리 변수 정의
  int duration, distance;

  //
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(1000);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = (duration/2) / 29.1;
  if (distance >= 200 || distance <= 0) {
    Serial.println("Out of range");
  } else {
```

```
    Serial.print(distance);
    Serial.println(" cm");
  }
  delay(500);
}
```

41.2 pulseIn 함수 사용 설명

위의 코드에는 duration = pulseIn(echoPin, HIGH); 라는 함수를 사용합니다.
직관적인 이해는 echoPin 으로 지정된 포트의 값이 HIGH 신호로 변경되기까지의 시간을 반환하는 함수입니다.
함수 선언 모체는 아래와 같습니다.

unsigned long pulsein(uint8_t pin, uint8_t state);
unsigned long pulseIn(uint8_t pin, uint8_t state, unsigned long timeout);

timeout 이 지정되지 않으면 기본값은 1,000,000 usec 입니다.

usec 설명

1 sec 는 모두가 아는 1 초입니다.
1 msec 는 1 초의 1,000 입니다.

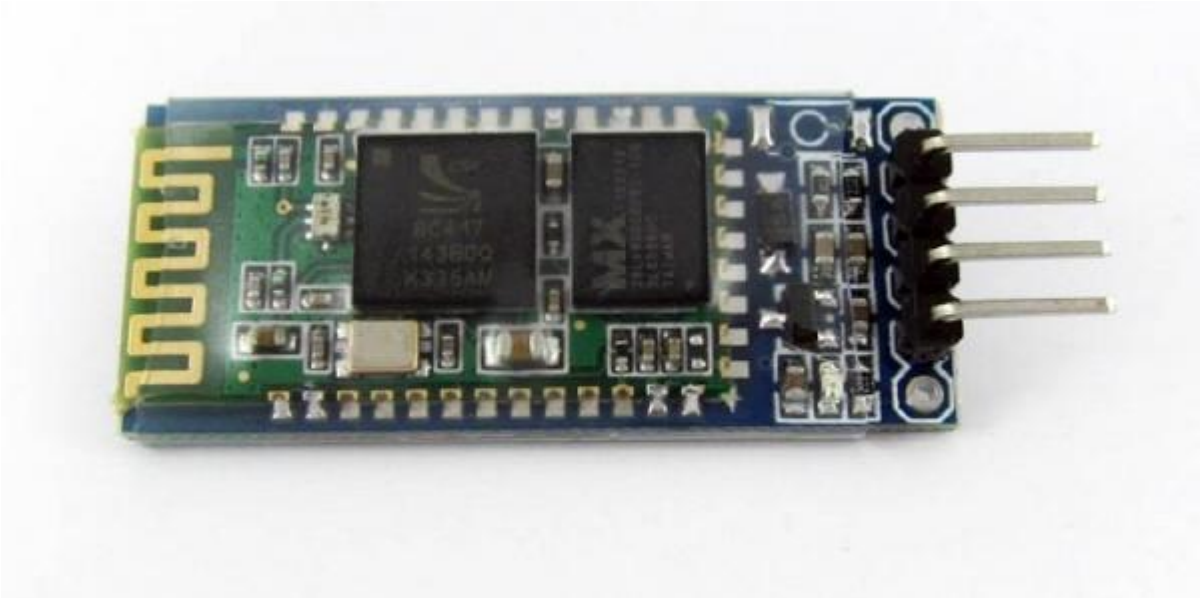
41.3 스케치 NEWPING 라이브러리 사용 예제

아두이노 사이트에서 NewPing 이라는 라이브러리 배포를 합니다. 초음파 센서 사용시 간단히 몇 줄로 구현 할 수 있습니다.
http://playground.arduino.cc/Code/NewPing

```
#include <NewPing.h>


#define TRIGGER_PIN  12
#define ECHO_PIN     11
#define MAX_DISTANCE 200

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

void setup() {
Serial.begin(115200);
}

void loop() {
delay(50);
int uS = sonar.ping();
Serial.print("Ping: ");
Serial.print(uS / US_ROUNDTRIP_CM);
Serial.println("cm");
}
```

42> 블루투스 HC-06 슬레이브 모듈

블루투스 통신 모듈입니다. HC-06 슬레이브 전용 모듈입니다.
블루투스(Bluetooth)는 휴대폰, 노트북, 헤드폰, 스피커, GPS, 등의 휴대기기를 서로 연결해 정보를 교환하는 근거리 무선 기술입니다.
대부분 10 미터 안팎의 근거리에서 저전력 무선 연결이 필요할 때 사용 됩니다.
블루투스 통신 기술을 사용한 제품들은 실생활에 무수히 많습니다.
가령, 블루투스 헤드폰을 사용하면 거추장스러운 케이블 없이도 주머니 속의 MP3 플레이어의 음악을 들을 수 있다.
블루투스 통신기술은 1994 년 휴대폰 공급업체인 에릭슨(Ericsson)이 시작한 무선 기술 연구를 바탕으로, 1998 년 에릭슨, 노키아, IBM, 도시바, 인텔 등으로 구성된 '블루투스 SIG(Special Interest Group)'를 통해 본격적으로 개발됐다. 이후 블루투스 SIG 회원은 급속도로 늘어나 2010 년 말 기준 전세계 회원사가 13,000 여 개에 이른다



블루투스 페어링 연결 기본 암호는 1234 입니다. 또는 0000 입니다.
추가 정보가 필요할 시에는 모듈의 STATE / WAKEUP 사용 가능합니다.

- 2.4GHz 안테나 내장
- EDR 블루투스 2.0, 2Mbps - 3Mbps
- 외부 8Mbit FLASH
- 3.3V ~5V 사용 가능. (최대 7V)

- 표준 HCI 포트 (UART)
- 옵션 PIO 제어
- SMD 배치 프로세스로 모듈
- 디지털 2.4GHz 무선 송신
- CSR BC04 블루투스 칩 기술
- 블루투스 클래스 2 전력 레벨
- RoHS 규제 절차
- 보관 온도: -40 +85 도, 작동 온도: -25 으로 75 도
- 외형 (27mm × 13mm × 2mm)

아두이노와의 프로그래밍에서는 소프트웨어 시리얼 라이브러리 또는 아두이노 포트의 하드웨어 UART 포트 **RX (D0), TX (D1)** 연결하여 사용 가능합니다.

아두이노 우노 R3 보드와 HC-06 블루투스 모듈 연결 합니다. 연결 방법은 아래의 41.3 글을 참조 합니다.

42.1 스마트폰 연동

스마트폰에서 구글 앱 스토어 접속 후 공개 프로그램 BlueTerm 검색, 설치 해줍니다.
블루텀 사이트:
https://play.google.com/store/apps/details?id=es.pymasde.blueterm
블루텀 앱을 안드로이드 스마트폰에 설치 후 블루투스 주변기기 검색을 해봅니다.
블루텀 소스코드는 공개입니다.(출처: http://pymasde.es/blueterm/)

아두이노 보드에 연결된 HC-06 장치가 검색되면 페어링을 해줍니다. 페어링시의 암호는 1234 입니다.
블루투스 모듈의 LED 깜박일 경우는 연결이 안된 상태입니다. 연결이 되었을 경우는 모듈 제조사에 따라 LED 항상 On 또는 통신 데이터 전송일 경우에만 점등됩니다.
연결이 되면 양방향 시리얼 통신이 가능합니다.
블루텀에서 입력한 문자는 바로 아두이노 보드에서 볼 수 있습니다.

42.2 PC 연동

PC 에서의 블루투스 HC-06 과 페어링을 하기 위해서는 사용하는 PC 의 블루투스의 드라이버를 되도록 제조사에서 배포하는 최신 버전으로 업데이트 해줍니다.
물론 윈 7, 윈 8 OS 에서 설정 되는 기본 블루투스 드라이버에서도 제대로 되는 경우도 있습니다.
블루투스 설정 도움말 사이트:
http://windows.microsoft.com/ko-kr/windows7/add-a-bluetooth-enabled-device-to-your-computer

42.3 블루투스 시리얼 통신

아두이노와 소프트웨어 시리얼 사용 방법.
블루투스 모듈의 GND 와 VCC 3.3V (5V) 연결해줍니다.
블루투스 모듈의 RX / TX 와 아두이노 보드의 D3, D2 핀에 연결 해 줍니다.

와이어링:

HC-06	아두이노 Uno
VCC	5V or 3V3
GND	GND
RX	D3
TX	D2

스케치 IDE 에서 아래의 예제 코드를 업로드 후 시리얼 모니터 창을 엽니다.
제대로 연결된 상태라면 스마트폰에서의 입력 그대로 시리얼 모니터 창에 나옵니다.
반대로 시리얼 모니터 창에서의 입력을 하면 스마트폰의 BlueTerm 화면에 문자가 표시됩니다.

// 소프트웨어 시리얼을 사용 하도록 합니다.

예제코드)

```
/*
블루투스를 사용하기 위한 준비를 합니다.
2 번은 TXD, 3 번은 RXD 입니다.
데이터를 주고 받을 핀 번호 입니다.
*/

#include <SoftwareSerial.h>

SoftwareSerial bt(2, 3);

void setup()
{
  Serial.begin(9600); // 시리얼 통신 초기화
  bt.begin(9600); // 블루투스를 사용하기 위해 초기화
  Serial.println("ready");
}

void loop()
{
  if(bt.available()) //블루투스를 통해 데이터가 들어오면
  {
    Serial.write("phone : ");
    Serial.write(bt.read()); //시리얼 모니터로 받은 내용 출력
    Serial.println();
  }
  if(Serial.available()) //시리얼 모니터에서 데이터를 보내면
  {
```

```
        bt.write("PC : ");
        bt.write(Serial.read()); //스마트폰으로 받은 데이터 출력
        bt.println();
    }
}
```

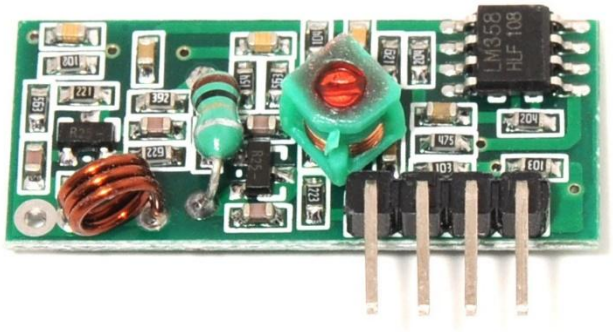
43 아두이노 무선통신 RF 433/315 MHz 송/수신 장치

무선통신이 가능한 433 MHZ Receiver & Transmitter 모듈입니다.

원격으로 송신 모듈과 수신 모듈 사이에 통신을 할 수 있습니다.
즉, 데이터 전송이 가능합니다. 데이터의 형태는 BYTE 이므로 문자열을 비롯한 많은 형태가 가능합니다.
모듈의 송/수신 거리는 장애물이 없을 경우 90m 이상입니다.

필요한 준비물은 아두이노 우노 보드 2 개가 필요합니다. 송신기를 장착한 MCU 보드와 수신기를 장착한 MCU 보드 각각의 송/수신 코드가 업로드 되어 테스트 됩니다.
물론, 코드 수정하여 송신, 수신 코드를 병합하여 테스트도 가능합니다.
본 키트에서는 아두이노 나노 V3 보드와 아두이노 우노 R3 보드를 사용하여 테스트 합니다.

수신 모듈



송신모듈 (Transmitter module)

공개된 스케치 라이브러리를 사용 합니다.
http://www.igameplus.com/pds/

무선 송신 코드입니다.

```
#include <VirtualWire.h>
char *controller;
void setup() {
  pinMode(13,OUTPUT);
   vw_set_ptt_inverted(true); //
```

```
  vw_set_tx_pin(12);
  vw_setup(4000);// speed of data transfer Kbps
}

void loop() {
  controller="1"   ;
  vw_send((uint8_t *)controller, strlen(controller));
  vw_wait_tx(); // Wait until the whole message is gone
  digitalWrite(13,1);
  delay(2000);
  controller="0"   ;
  vw_send((uint8_t *)controller, strlen(controller));
  vw_wait_tx(); // Wait until the whole message is gone
  digitalWrite(13,0);
  delay(2000);
}
```

The D13 LED On the arduino board must be turned ON when received character '1' and Turned Off when received character '0'

```
 #include <VirtualWire.h>

 void setup()
 {
       vw_set_ptt_inverted(true); // Required for DR3100
       vw_set_rx_pin(12);
       vw_setup(4000);   // Bits per sec
       pinMode(13, OUTPUT);

       vw_rx_start();           // Start the receiver PLL running
 }
```

```
void loop()
{
	uint8_t buf[VW_MAX_MESSAGE_LEN];
	uint8_t buflen = VW_MAX_MESSAGE_LEN;

	if (vw_get_message(buf, &buflen)) // Non-blocking
	{
		if(buf[0]=='1')
		{
			digitalWrite(13,1);
		}
		if(buf[0]=='0')
		{
			digitalWrite(13,0);
		}
	  }
}
```

44> 프라임 키트 부품 보관 상자

Plastic Box – Bi level & Partition

키트 구성 항목을 보관 할 수 있는 플라스틱 상자입니다.

내부는 2 단으로 구성되어 있습니다. 파티션 가능 구조물이 있어 분류 별로 보관 할 수 있습니다.

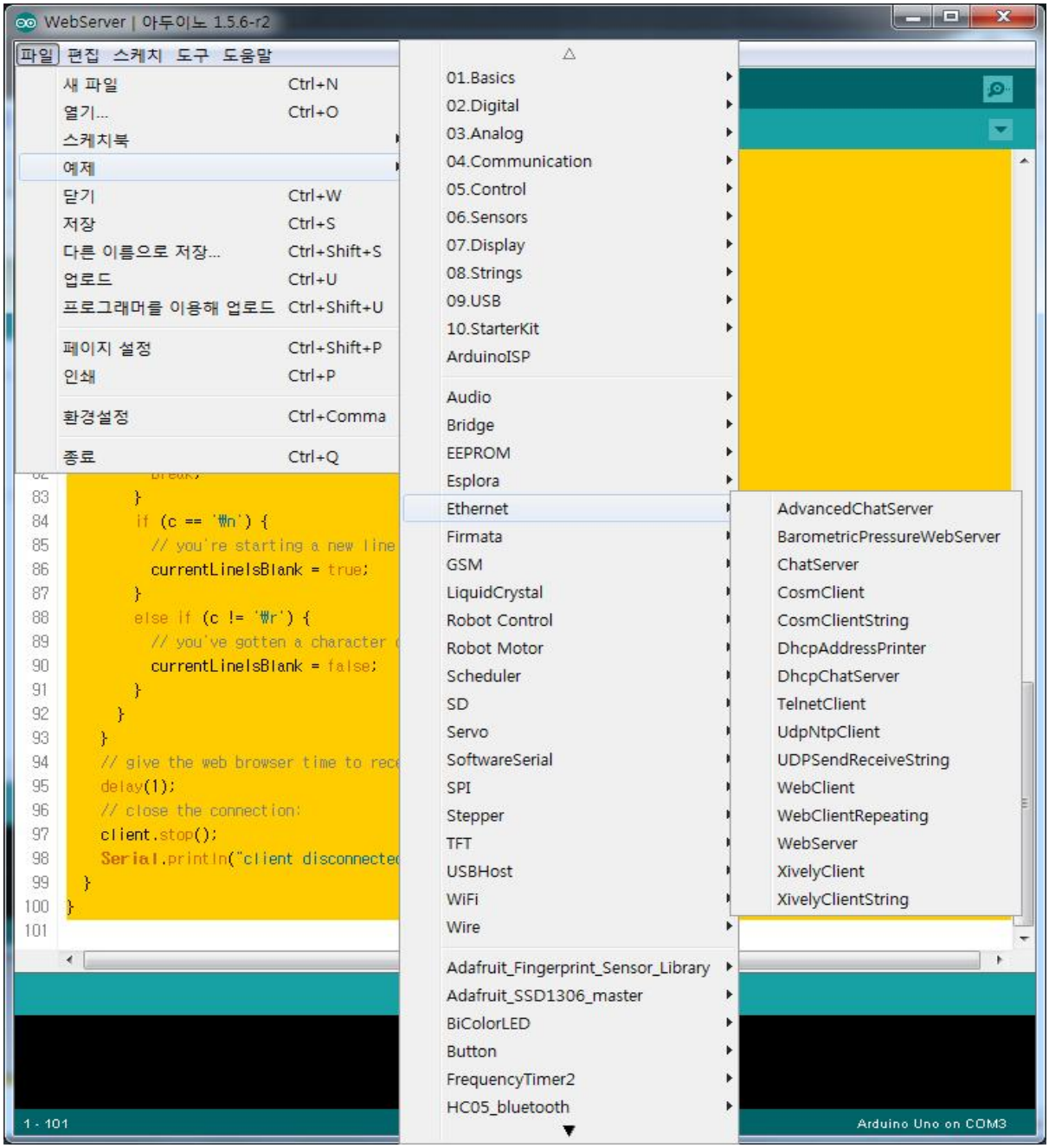
45 아두이노 네트워크 프로그래밍

아두이노에서 네트워크 프로그래밍을 가능하게 하는 이더넷 쉴드 입니다.
쉴드 형태 또는 모듈들이 있습니다.
키트에서 사용될 장치는 Arduino Ethernet Shield W5100 입니다.
W5100 이더넷 칩셋은 100 Mbps 속도입니다.

아두이노 스케치 IDE 프로그램에는 Ethernet 사용 가능한 라이브러리가 포함 되어 있습니다.
위의 이미지와 같은 W5100 이더넷 쉴드를 사용하면 그대로 예제 코드를 가져와서 사용할 수 있습니다.
본 모듈은 마이크로 SD 카드 와 이더넷 칩셋이 가능한 쉴드입니다.

스케치 IDE 의 메뉴→예제→Ethernet→ 다수의 이더넷 예제가 포함되어 있습니다.
이더넷 상세 라이브러리의 기능을 몰라도 네트워크 접속/해제, 읽기/쓰기(송신/수신)를 할 수 있습니다.

이더넷 예제 코드는 스케치 라이브러리에서 아래의 이미지처럼 많은 예제가 있습니다.



아두이노 이더넷 네트워크 예제중에
WebClient 라는 예제 항목을 열기 합니다.

이더넷 쉴드 W5100 사용하는 경우는 쉴드 장착만 하면 아래의 코드를 그대로 사용 가능합니다. 내부 SPI 연결 되어 있는 상태입니다.
이더넷 쉴드의 RJ45 케이블(네트워크 케이블)이 인터넷 공유기에 물려 있는 상태로 테스트합니다.

코드의 작동은 인터넷 연결이 되어 있는 경우에는 구글 홈페이지의 내용을 시리얼 포트로 출력해 줍니다.
DHCP 방식 공유기 사용중인 경우 아래의 코드 중 이해해야 될 부분은

```
// 이더넷 시작
// DHCP 방식으로 ip 할당을 받습니다.
if (Ethernet.begin(mac) == 0) {
        Serial.println("Failed to configure Ethernet using DHCP");
        // no point in carrying on, so do nothing forevermore:
        // try to congifure using IP address instead of DHCP:
        // 만약 DHCP ip 할당이 실패한 경우 ip 변수를 사용하여 ip 지정.
        Ethernet.begin(mac, ip);
  }


/*
  Web client

 This sketch connects to a website (http://www.google.com)
 using an Arduino Wiznet Ethernet shield.

 Circuit:
 * Ethernet shield attached to pins 10, 11, 12, 13

 created 18 Dec 2009
 by David A. Mellis
 modified 9 Apr 2012
 by Tom Igoe, based on work by Adrian McEwen

 */

#include <SPI.h>
#include <Ethernet.h>
```

```
// Enter a MAC address for your controller below.
// Newer Ethernet shields have a MAC address printed on a sticker on the shield
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
// if you don't want to use DNS (and reduce your sketch size)
// use the numeric IP instead of the name for the server:
//IPAddress server(74,125,232,128);  // numeric IP for Google (no DNS)
char server[] = "www.google.com";      // name address for Google (using DNS)

// Set the static IP address to use if the DHCP fails to assign
IPAddress ip(192, 168, 0, 177);

// Initialize the Ethernet client library
// with the IP address and port of the server
// that you want to connect to (port 80 is default for HTTP):
EthernetClient client;

void setup() {
  // Open serial communications and wait for port to open:
  Serial.begin(9600);

// while (!Serial) {
  // ; // wait for serial port to connect. Needed for Leonardo only
  }

  // start the Ethernet connection:
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP");
    // no point in carrying on, so do nothing forevermore:
    // try to congifure using IP address instead of DHCP:
    Ethernet.begin(mac, ip);
  }
  // give the Ethernet shield a second to initialize:
  delay(1000);
  Serial.println("connecting...");
```

```
  // if you get a connection, report back via serial:
  if (client.connect(server, 80)) {
    Serial.println("connected");
    // Make a HTTP request:
    client.println("GET /search?q=arduino HTTP/1.1");
    client.println("Host: www.google.com");
    client.println("Connection: close");
    client.println();
  }
  else {
    // kf you didn't get a connection to the server:
    Serial.println("connection failed");
  }
}


// loop 함수의 내용은 네트워크 포트에 데이터가 있을 경우 시리얼로 출력해 줍니다.
void loop()
{
  // if there are incoming bytes available
  // from the server, read them and print them:
  if (client.available()) {
    char c = client.read();
    Serial.print(c);
  }

  // if the server's disconnected, stop the client:
  if (!client.connected()) {
    Serial.println();
    Serial.println("disconnecting.");
    client.stop();

    // do nothing forevermore:
```

```
    while (true);
  }
}
```

46 SD 카드 읽기/쓰기

아두이노 스케치 IDE 기본 라이브러리에는 SD 카드 읽기/쓰기 라이브러리가 있습니다. SD 카드 라이브러리를 사용하기 위해서는 반드시 헤더 파일을 선언해 주어야 합니다.

```
#include <SPI.h>
#include <SD.h>
```

스케치 IDE 메뉴 예제 -> SD-> listFiles 열기 합니다.
예제 코드는 SD 카드의 루트 디렉터리 하위 포함 모든 파일 목록을 시리얼 포트로 출력해 줍니다.

```
/*
  Listfiles

 This example shows how print out the files in a
 directory on a SD card

 The circuit:
 * SD card attached to SPI bus as follows:
 ** MOSI - pin 11
 ** MISO - pin 12
 ** CLK - pin 13
 ** CS - pin 4
```

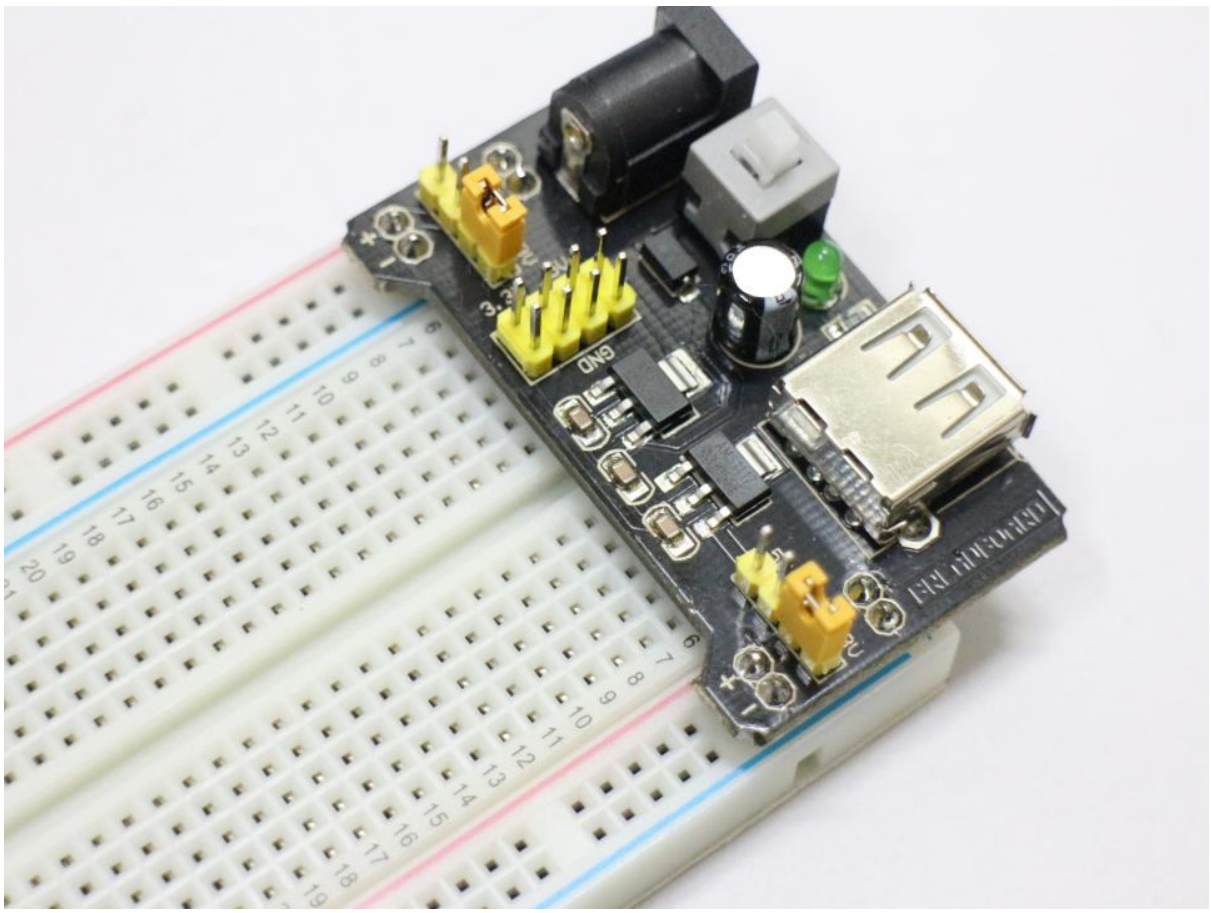
```
 created   Nov 2010
 by David A. Mellis
 modified 9 Apr 2012
 by Tom Igoe
 modified 2 Feb 2014
 by Scott Fitzgerald

 This example code is in the public domain.
 */

#include <SPI.h>
#include <SD.h>

// 파일 클래스 글로벌 변수 선언.
File root;

void setup()
{
  // Open serial communications and wait for port to open:
  Serial.begin(9600);
  // while (!Serial) {
  //   ; // wait for serial port to connect. Needed for Leonardo only
  // }

  Serial.print("Initializing SD card...");
  // On the Ethernet Shield, CS is pin 4. It's set as an output by default.
  // Note that even if it's not used as the CS pin, the hardware SS pin
  // (10 on most Arduino boards, 53 on the Mega) must be left as an output
  // or the SD library functions will not work.
  pinMode(10, OUTPUT);

  if (!SD.begin(4)) {
    Serial.println("initialization failed!");
    return;
```

```
  }
  Serial.println("initialization done.");

  root = SD.open("/");

  printDirectory(root, 0);

  Serial.println("done!");
}

void loop()
{
  // nothing happens after setup finishes.
}

void printDirectory(File dir, int numTabs) {
   while(true) {

     File entry =  dir.openNextFile();
     if (! entry) {
       // no more files
       break;
     }
     for (uint8_t i=0; i<numTabs; i++) {
       Serial.print('₩t');
     }
     Serial.print(entry.name());
     if (entry.isDirectory()) {
       Serial.println("/");
       printDirectory(entry, numTabs+1);
     } else {
       // files have sizes, directories do not
       Serial.print("₩t₩t");
       Serial.println(entry.size(), DEC);
```

```
        }
        entry.close();
    }
}
```

47 BREADBOARD POWER BOARD 2 LOAD 3V3,5V

브레드보드 전원 공급장치 입니다. 입고 순서에 따라 검정색 PCB 형태와 약간의 PCB 형태가 다를 수도 있습니다.



브레드보드와 결합 예시 이미지

USB 및 DC 어댑터의 전원을 받아 브레드보드에 전원을 공급하여 주는 장치입니다.
브레드보드에 전원공급 모듈로서 5V/3.3V 를 동시에 사용 가능한 제품입니다.
출력은 5V/3.3V 전원 선택 가능하며, 개별 전원 공급 가능한 제품입니다.
각각의 오른쪽, 왼쪽 상단에 3V3, 5V 전원 설정 가능한 스위치가 있습니다.
전원 연결은 USB A Type & DC 입니다.

DC 입력 연결은 7V~12V 사용할 수 있습니다.

브레드보드의 앞 번호 부분에 장착해야 +, - 방향이 맞습니다. 주의하시기 바랍니다.

48 프라임 키트 구성 항목입니다.

품목	개수
➢ Arduino UNO R3 Board	1
➢ High quality and large bread board	1
➢ RFID module	1
➢ RFID IC Keychain	1
➢ RFID Non-contact type IC card	1
➢ I2C LCD 1602 module	1
➢ 5v Relay 1 Channel	1
➢ DS1302 clock module	1
➢ Voice detection module	1
➢ Temperature and humidity module	1
➢ Level detection module	1
➢ 4 * 4 keypad module	1
➢ Three-color RGB module	1
➢ XY joystick	1
➢ Servo motor	1
➢ Stepper motor driver board	1
➢ Green LED	5
➢ Yellow LED	5

➢ Red LED	5
➢ 1 K ohm resistor	10
➢ 10 K ohm resistor	10
➢ 220 ohm resistor	10
➢ Buzzer	2
➢ Key module (with hat) 12x12mm	4
➢ Tilt Sensor/Switch	2
➢ LM35 sensor module	1
➢ Photo resistor	3
➢ Fire sensor, Flame Sensor	1
➢ Infrared receiver	1
➢ Adjustable potentiometer	1
➢ A digital control	1
➢ 4 digital tube 1 Digit 7-Segment	1
➢ 8 * 8 dot matrix module	1
➢ 74HC595N chip	1
➢ Infrared remote control	1
➢ Breadboard Jumper Cable	65
➢ Male to Female DuPont lines	10
➢ 9 volt battery snap	1
➢ USB Cable B-Type	1
➢ HC-SR04 Ultrasonic Sensor	1
➢ Plastic Box – Bi level & Partition	1

49 프라임 키트 플러스 구성 항목입니다.

네트워크 & 블루투스 통신 기능이 추가됩니다.

품목	개수
➢ Arduino Ethernet Shield W5100	1
➢ Arduino Uno Prototype Plate Shield	1
➢ Arduino Uno Sensor Shield V4	1
➢ Bluetooth HC-06 4 pin Slave	1
➢ Breadboard Power Board 2 Load 3V3,5V	1
➢ Breadboard Standard 400 points	1
➢ Breadboard Mini 170 points	2
➢ DuPont Male to Female Cable 40 pin	1
➢ DuPont Female to Female Cable 40 pin	1
➢ Plastic Kit Large Box	1

50프라임 키트 얼티밋 구성 항목입니다.

XBee 통신과 파일 저장 및 OLED 디스플레이 키트입니다.
XBee 통신 테스트를 위해 2 개의 모듈이 포함됩니다.

(구성항목 현재 미정의)

품목	개수
➢ XBee S2 Module	2
➢ XBee USB Adaptor	1
➢ XBee Shield	1
➢ OLED 0.96 inch 128x64 SPI Interface 7 Pin	1
➢ Micro SD Slot Module	1
➢ Arduino Nano V3 Board with USB Cable	1

부록

아두이노 스케치 IDE 사용하기

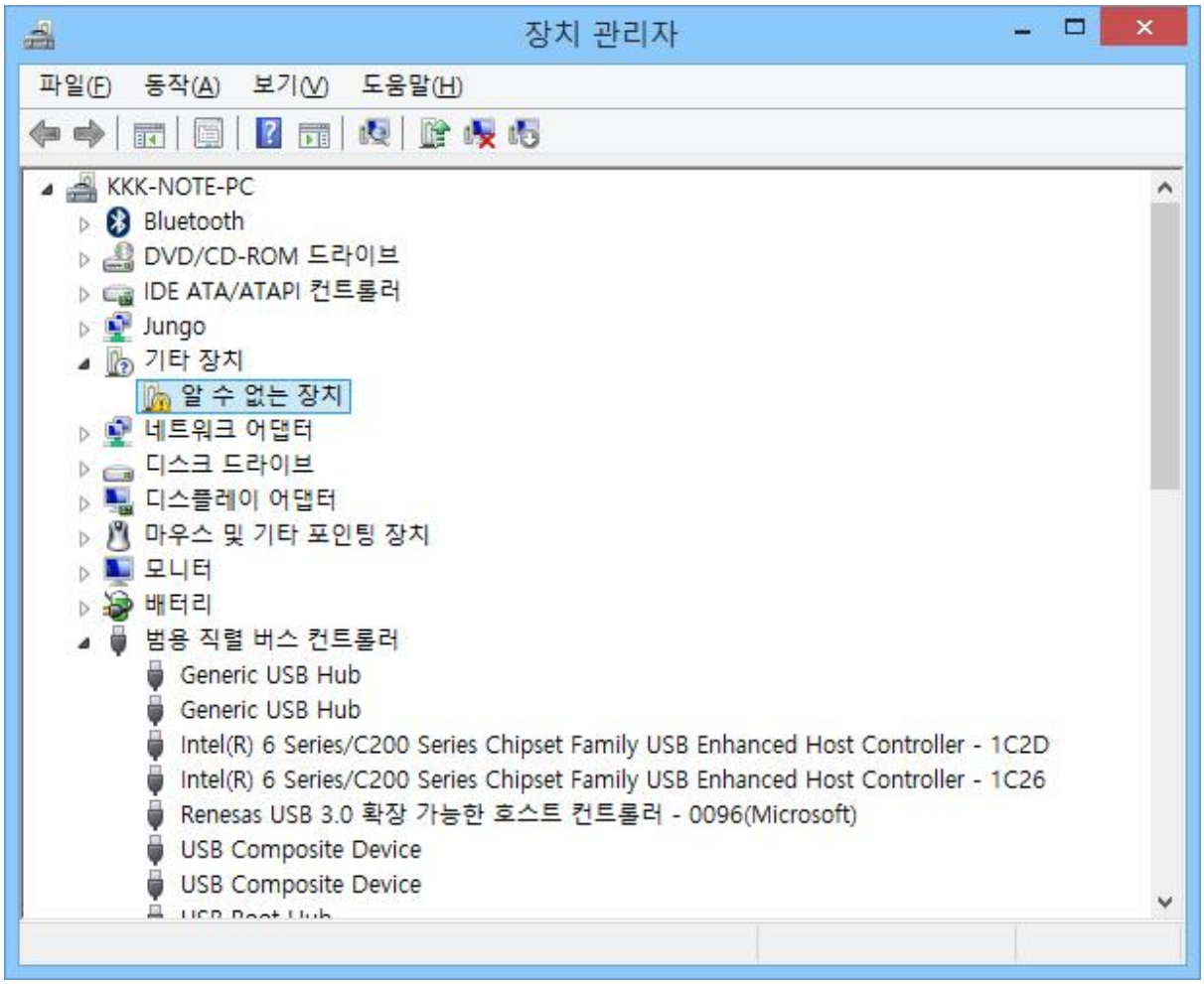
1 아두이노 우노 R3 보드 설치

아두이노 우노 R3 보드를 사용하기 위해서는 USB 연결 후 사용합니다.

PC 에서 사용하기 위해서는 드라이버를 설치해주어야 작동됩니다.
아두이노 우노를 비롯하여 대부분의 아두이노 보드들은 PC 와 연결 시 별도의 전원이 없이 USB 케이블 연결로만 작동 가능합니다.
USB 케이블을 통해 시리얼 통신 신호와 더불어 전원도 공급받아서 사용됩니다.
아두이노 우노 R3 보드는 B-Type USB 연결 케이블을 사용합니다.

1.1 아두이노 보드 드라이버 설치 방법

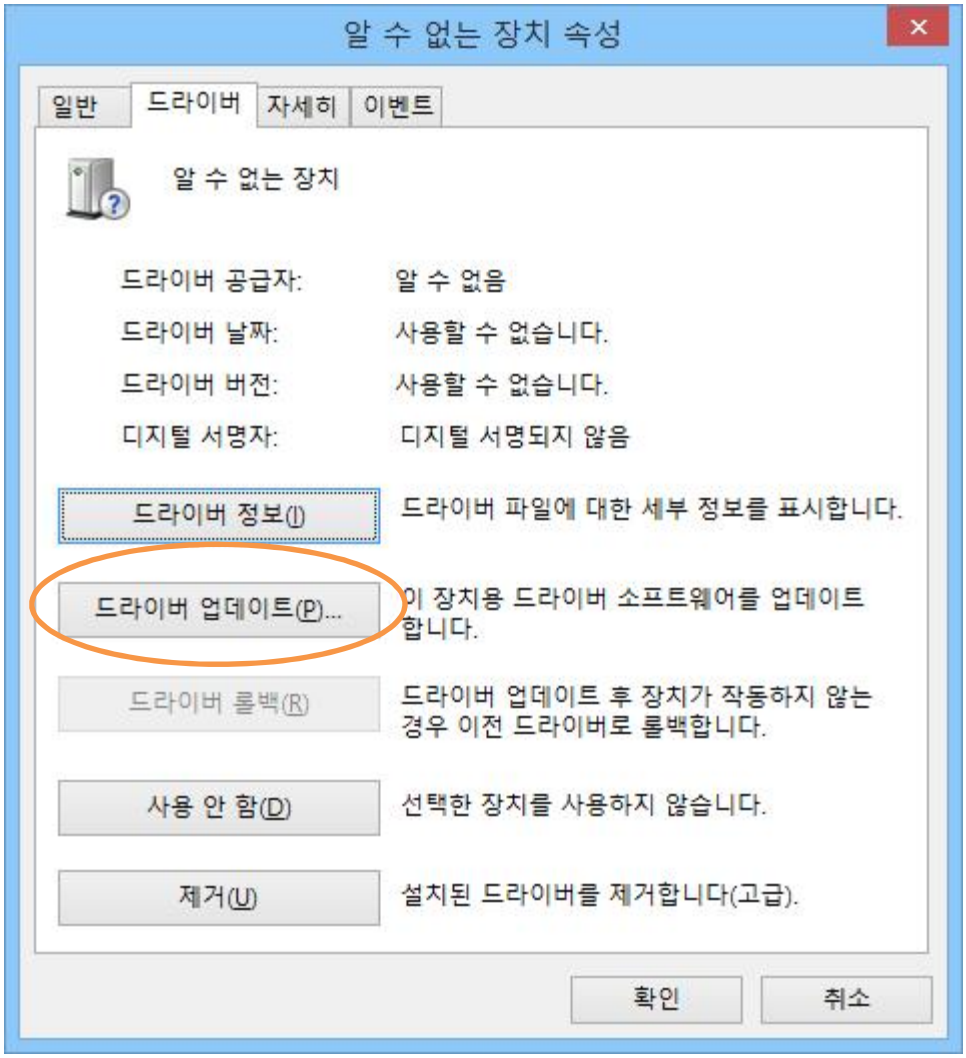
아두이노 보드와 PC 의 USB 포트에 연결 시 드라이버 설정이 안되어 있을 경우 드라이버를 설치 해주어야 합니다.

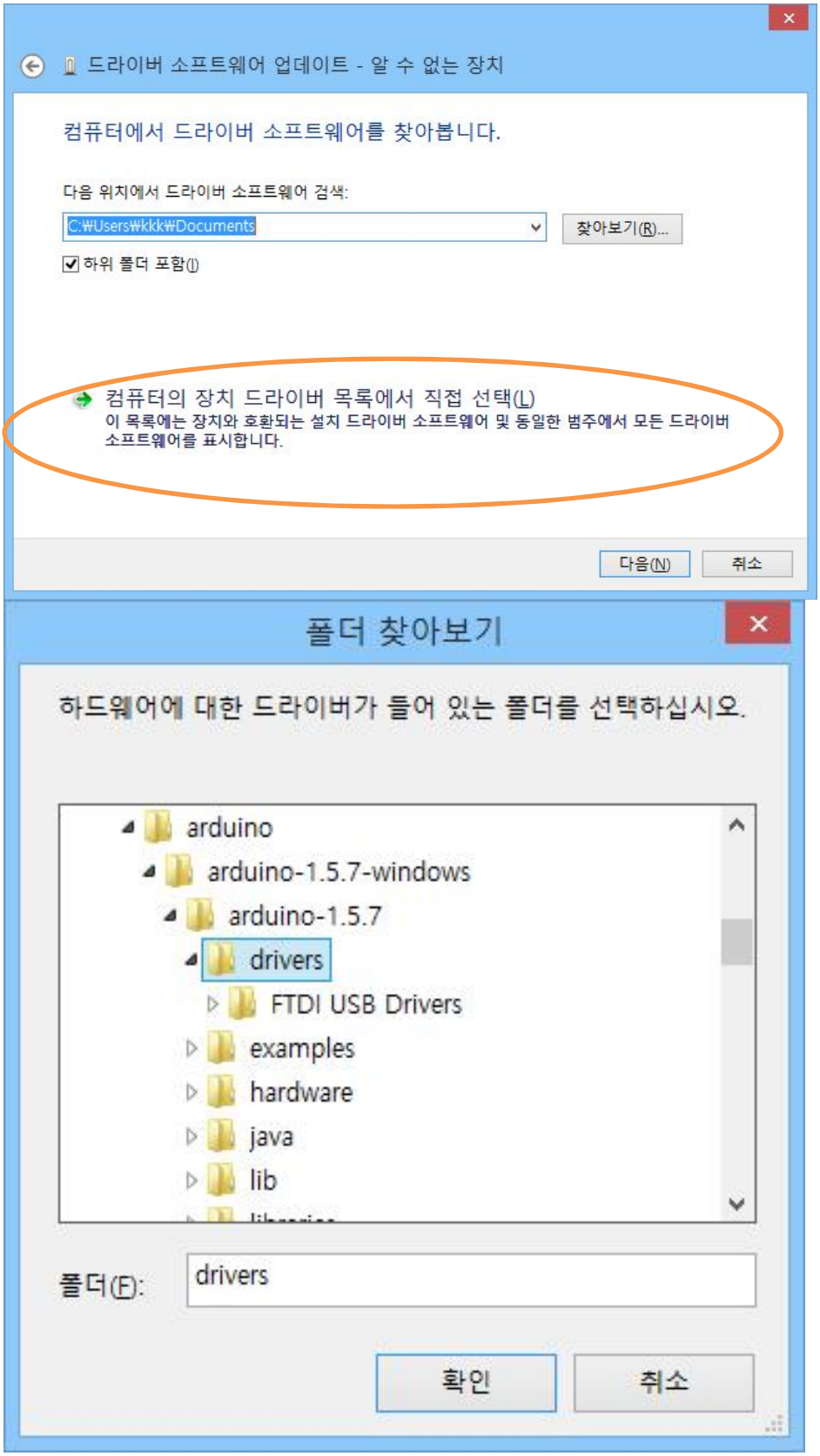


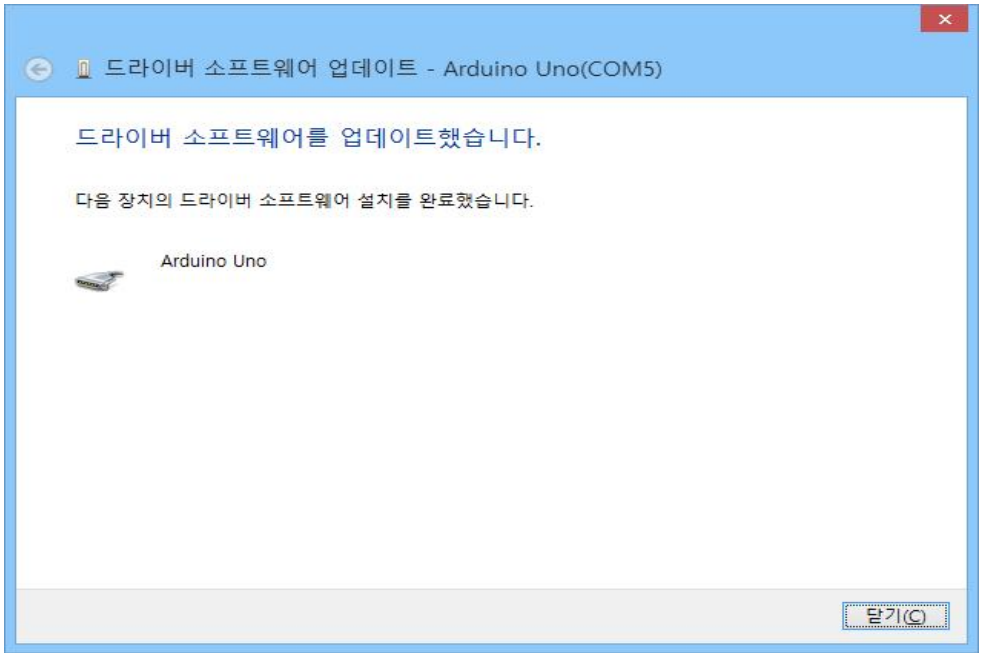
드라이버 파일은 http://arduino.cc/en/Main/Software 다운로드 받습니다.
다운받은 드라이버 파일을 압축 해제 후 하위 디렉터리의 ₩Arduino₩arduino-1.5.7-windows₩arduino-1.5.7₩drivers 라는 디렉터리에 관련된 파일들이 있습니다.

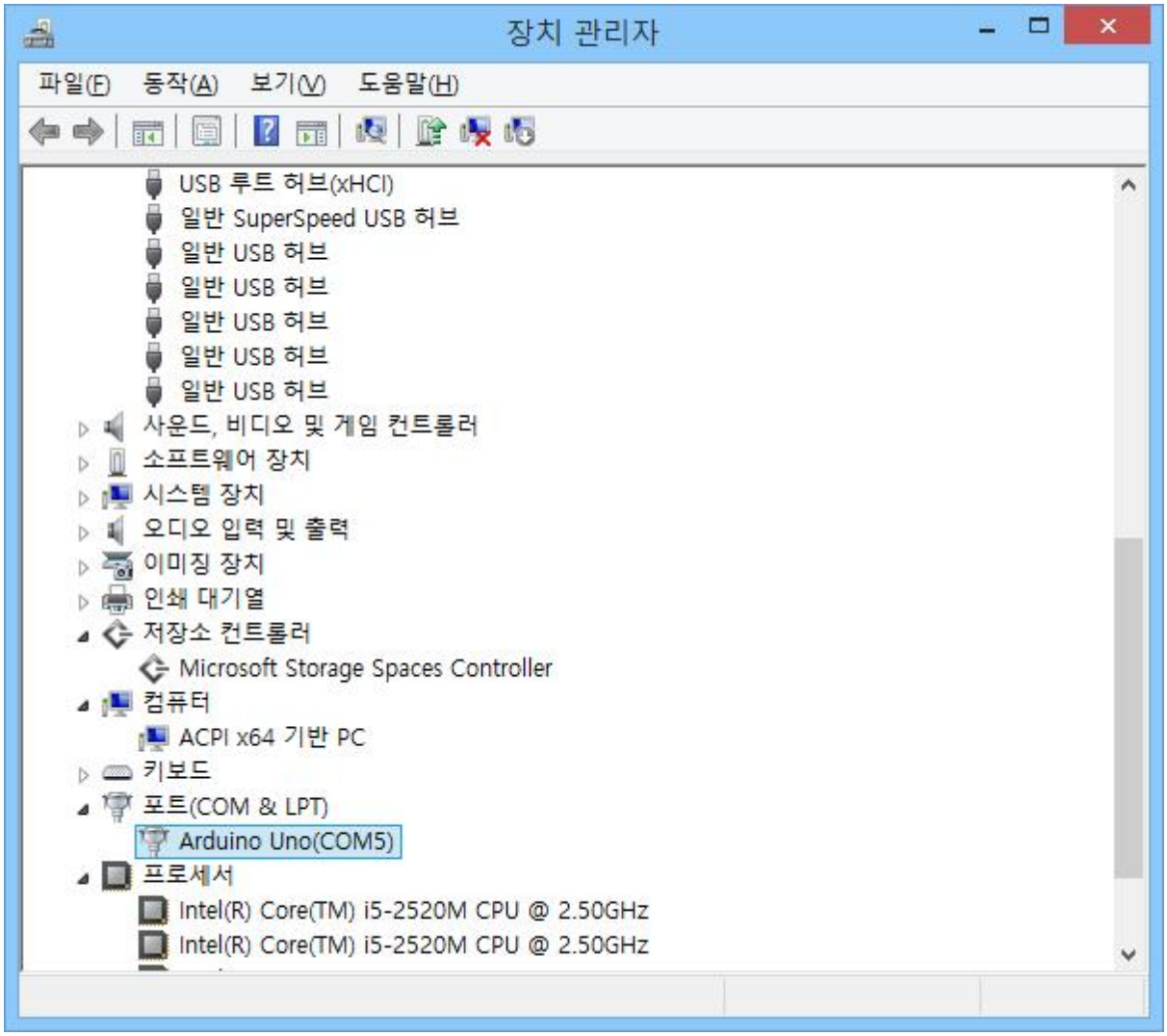
"드라이버 업데이트" 버튼을 눌러서 드라이버 파일이 있는 디렉터리를 지정 해줍니다

.




드라이버 소프트웨어 업데이트 - 알 수 없는 장치
컴퓨터에서 드라이버 소프트웨어를 찾아봅니다.
다음 위치에서 드라이버 소프트웨어 검색:
C:₩Users₩kkk₩Documents
찾아보기(R)...
하위 폴더 포함(I)
컴퓨터의 장치 드라이버 목록에서 직접 선택(L)
이 목록에는 장치와 호환되는 설치 드라이버 소프트웨어 및 동일한 범주에서 모든 드라이버 소프트웨어를 표시합니다.
다음(N)
취소
폴더 찾아보기
하드웨어에 대한 드라이버가 들어 있는 폴더를 선택하십시오.
arduino
arduino-1.5.7-windows
arduino-1.5.7
drivers
FTDI USB Drivers
examples
hardware
java
lib
폴더(F):
drivers
확인
취소


드라이버 소프트웨어 업데이트 - Arduino Uno(COM5)
드라이버 소프트웨어를 업데이트했습니다.
다음 장치의 드라이버 소프트웨어 설치를 완료했습니다.
Arduino Uno
닫기(C)



2 아두이노 스케치 프로그램 설치 & 설정.

아두이노 공식 사이트에서 스케치 프로그램을 다운로드 받습니다.

http://arduino.cc/en/Main/Software

현재 배포버젼은 1.5.7 입니다.

필요에 따라 다른 버전을 사용하여도 무방합니다.

Download

Arduino 1.5.7 (release notes):

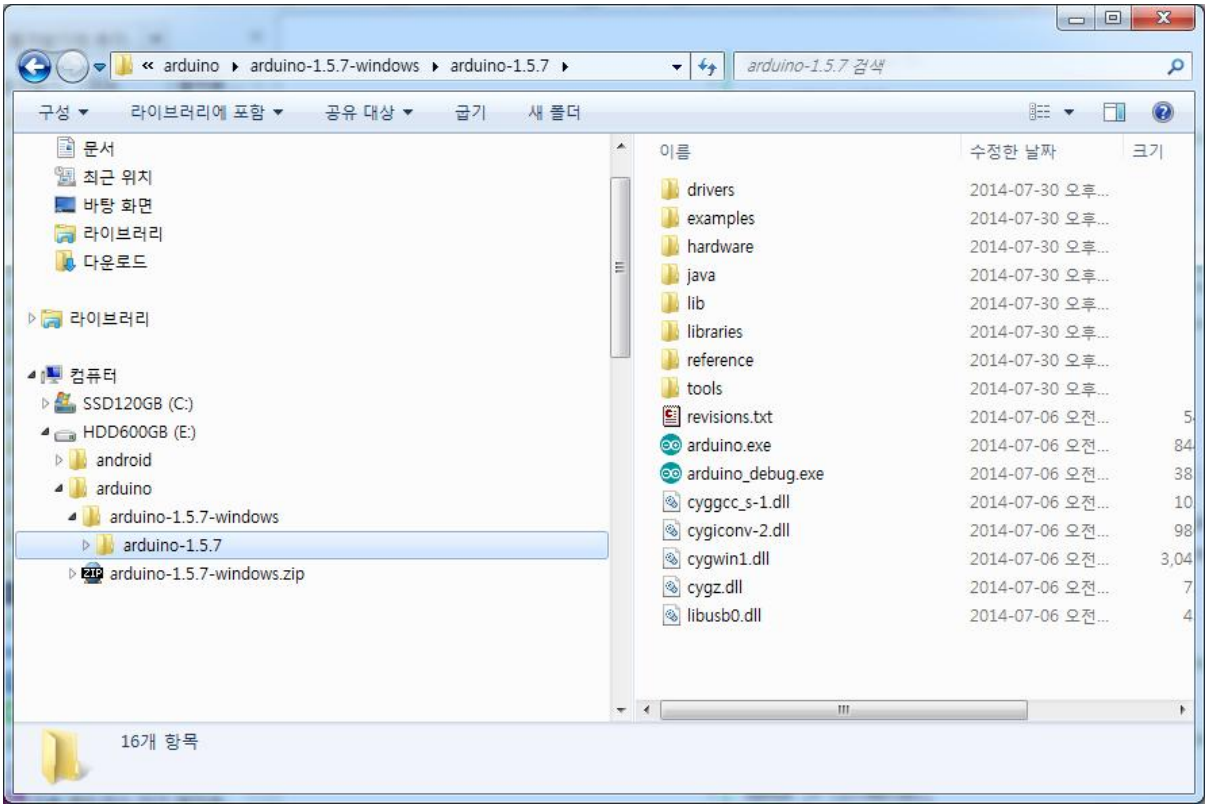
- Windows: Installer
- Windows: ZIP file (for non-administrator install)

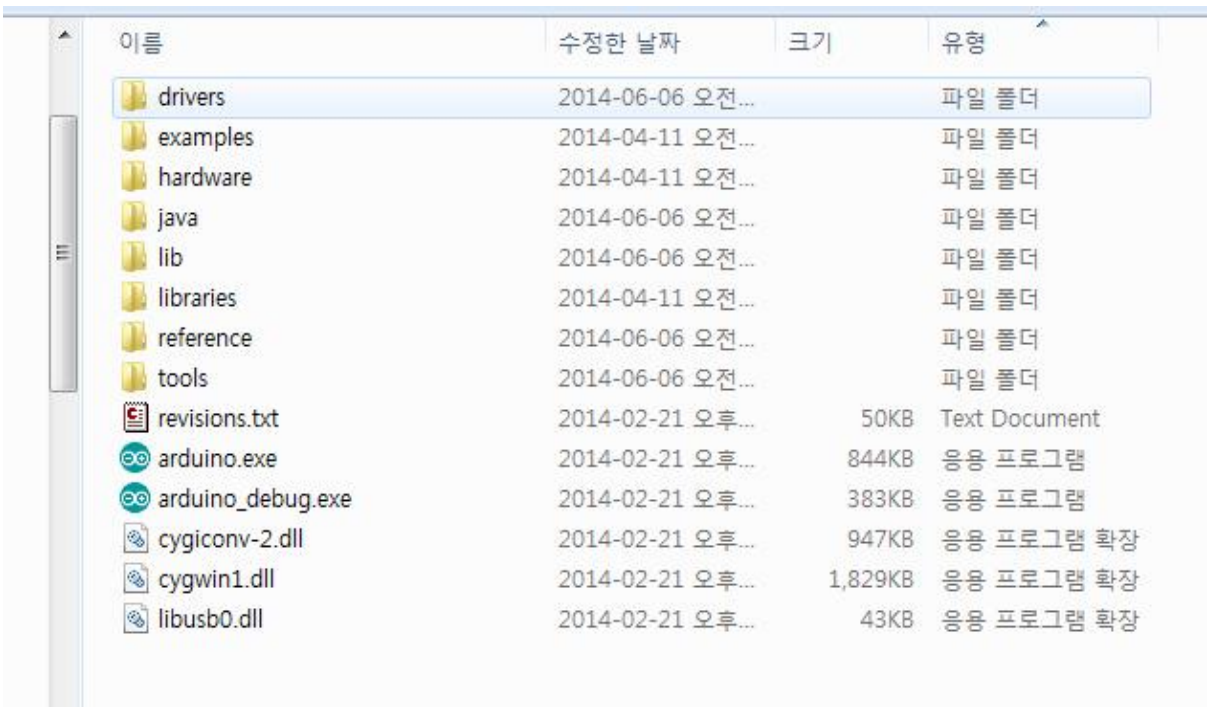
- Mac OS X: ZIP file for Java 6
- Mac OS X: ZIP file for Java 7. NOTE: the ZIP for Java 7 is experimental and it works only on OSX 10.7 or greater. If you experience problems running it please use the Java 6 version.
- Linux: 32 bit, 64 bit
- source

Windows: Installer → 윈도우 OS 에서 사용 가능한 설치 형식의 파일.
Windows: ZIP file → 다운로드 후 원하는 디렉터리에 압축 해제 후 사용.
Mac OS X: ZIP File for Java 6 → 애플 OS X 에서 사용 가능한 버전 입니다.
Mac OS X: Zip file for Java 7 → OSX 10.7 이상의 버전에서 사용 가능한 버전 입니다.
Linux: 32 비트, 64 비트 버전
Source: 스케치 IDE 프로그램 소스

Windows: Zip File 사용 방식의 파일을 다운로드 받아서 원하는 디렉터리에 압축 해제 후 사용합니다.
1.5.8 버전 압축 파일 형태의 배포 파일을 받아서 원하는 디렉터리 에 압축 해제하면 아래와 같은 형태의 디렉터리가 보입니다.
본 예제에서는 하드 디스크 E:₩ 드라이브의 arduino 라는 디렉터리를 생성 후 다운로드 받은 파일을 압축 해제 하였습니다.

E:₩arduino 디렉터리 생성
E:₩arduino 디렉터리에 다운로드 받은 arduino-1.5.8-windows.zip 파일 복사
arduino-1.5.8-windows.zip 파일을 압축 해제합니다.
아래와 같은 디렉터리가 나옵니다.
디렉터리 E:₩arduino₩arduino-1.5.7-windows₩arduino-1.5.7 에는 아두이노에 관련된 모든 파일들이 있습니다.





Arduino -> drivers 디렉터리(폴더)에 각종 윈도우 드라이버 파일들이 있습니다.

아두이노 구매 후 USB 케이블 연결 시 드라이버를 찾을 수 없는 경우 Arduino->drivers 디렉터리의 dpinst-amd64.exe (64 비트 윈도우 드라이버 설치) 실행합니다.

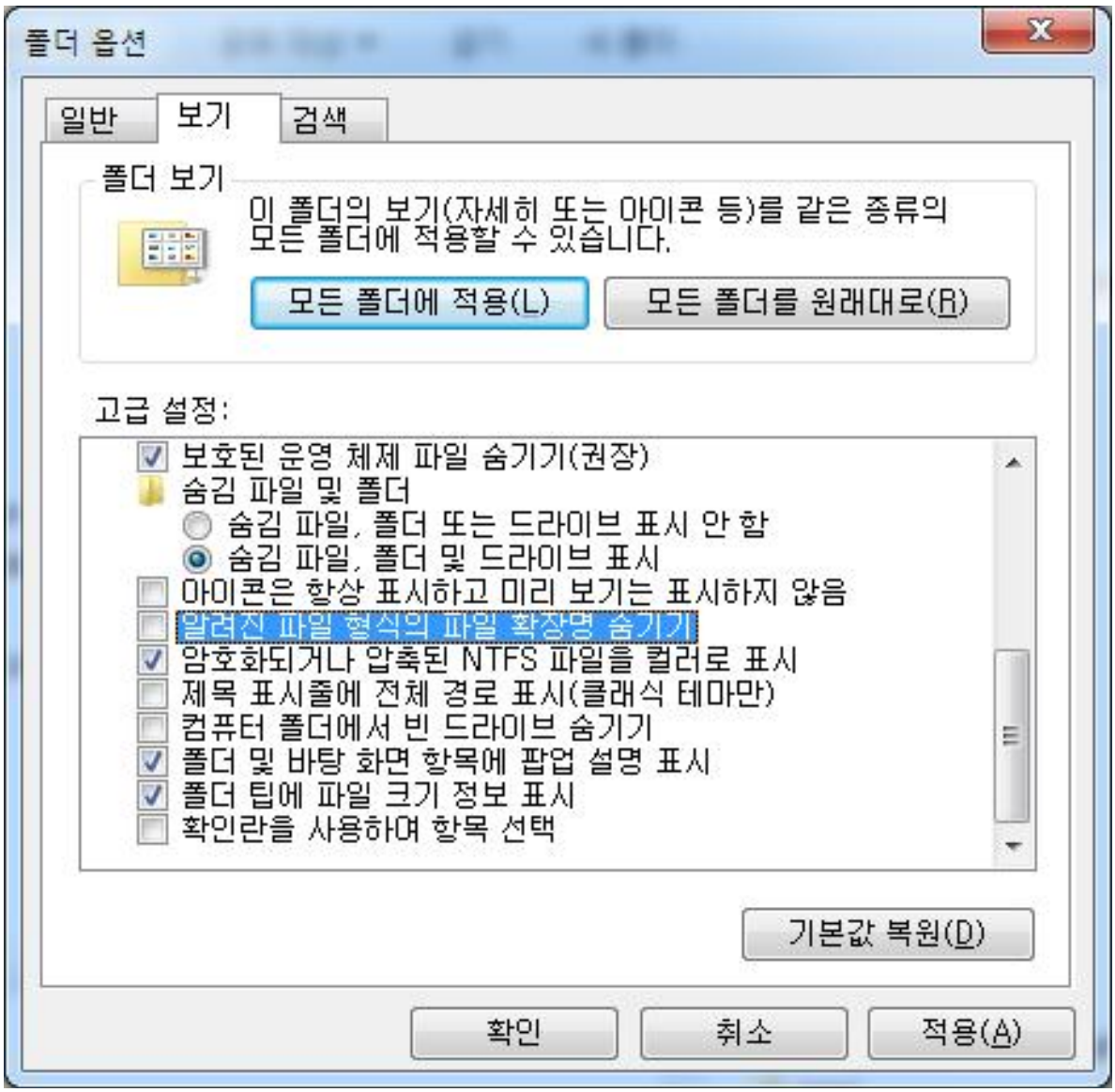
스케치 프로그램 실행 파일은 arduino.exe 입니다.

2.1 윈도우 탐색기에서 파일 확장자 보이도록 설정 방법

윈도우 XP, 윈도우 7, 윈도우 8 탐색기 설정 도구->폴더 옵션->보기(옵션 탭)" 에 "알려진 파일 형식의 파일 확장명 숨기기" 체크 버튼 설정 해제 되어 있어야 파일의 확장 명칭이 보입니다.

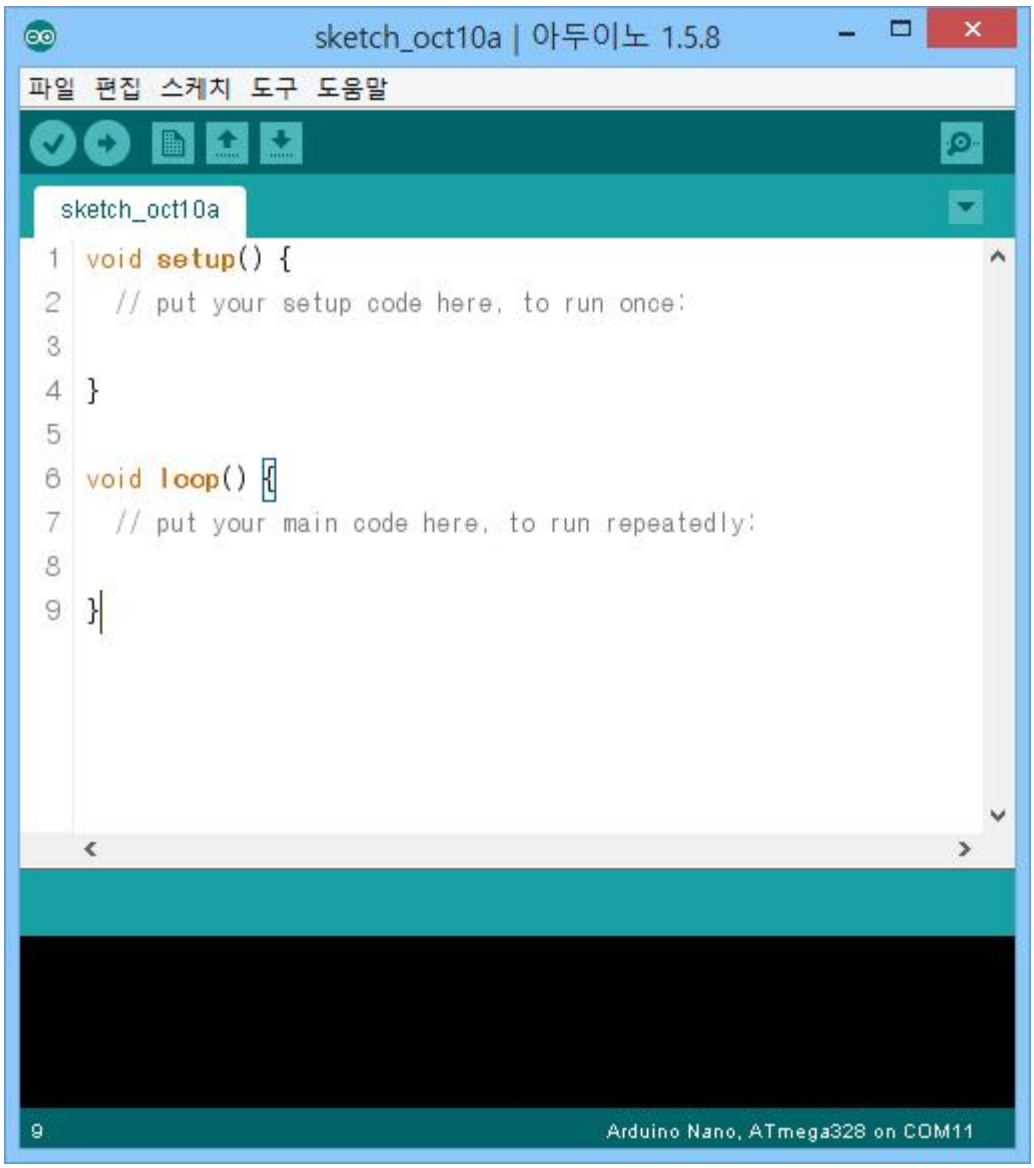
확장명 없어도 사용하기 불편함이 없으면 "확장명 숨기기" 옵션으로도 상관 없습니다.

그럼 위의 arduino.exe 라고 탐색기에서 표시되지 않습니다. arduino 라고만 보입니다.



2.2 스케치 기본 함수

스케치 IDE 파일 메뉴 ->"새 파일" 선택하면 새로운 스케치 IDE 창이 생성 되면서 아래의 코드가 나옵니다.



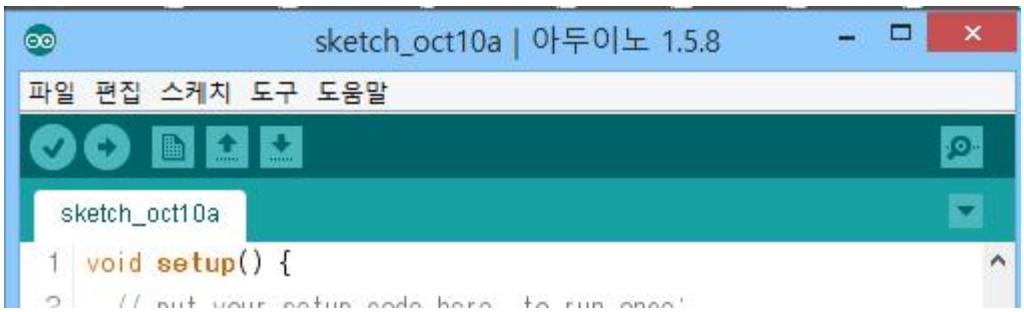
void setup() 함수는 아두이노 보드 부팅 시 1 회 호출되는 함수 입니다. 장착된 하드웨어 설정 및 코드 초기화를 할 수 있습니다.

void loop() 함수는 아두이노 보드 실행 시 매번 호출 되는 함수입니다.

아두이노 보드의 전원이 해제 되거나, 이상이 있을 경우를 제외 하고는 계속 실행되는 함수입니다.

2.3 스케치 프로그램 IDE 기본 기능

스케치 IDE 의 상단에는 많이 사용되는 기능의 버튼들이 있습니다.



 스케치 프로그램 컴파일 버튼. 코드 컴파일 기능.

 스케치 프로그램 업로드 버튼.

 새 파일

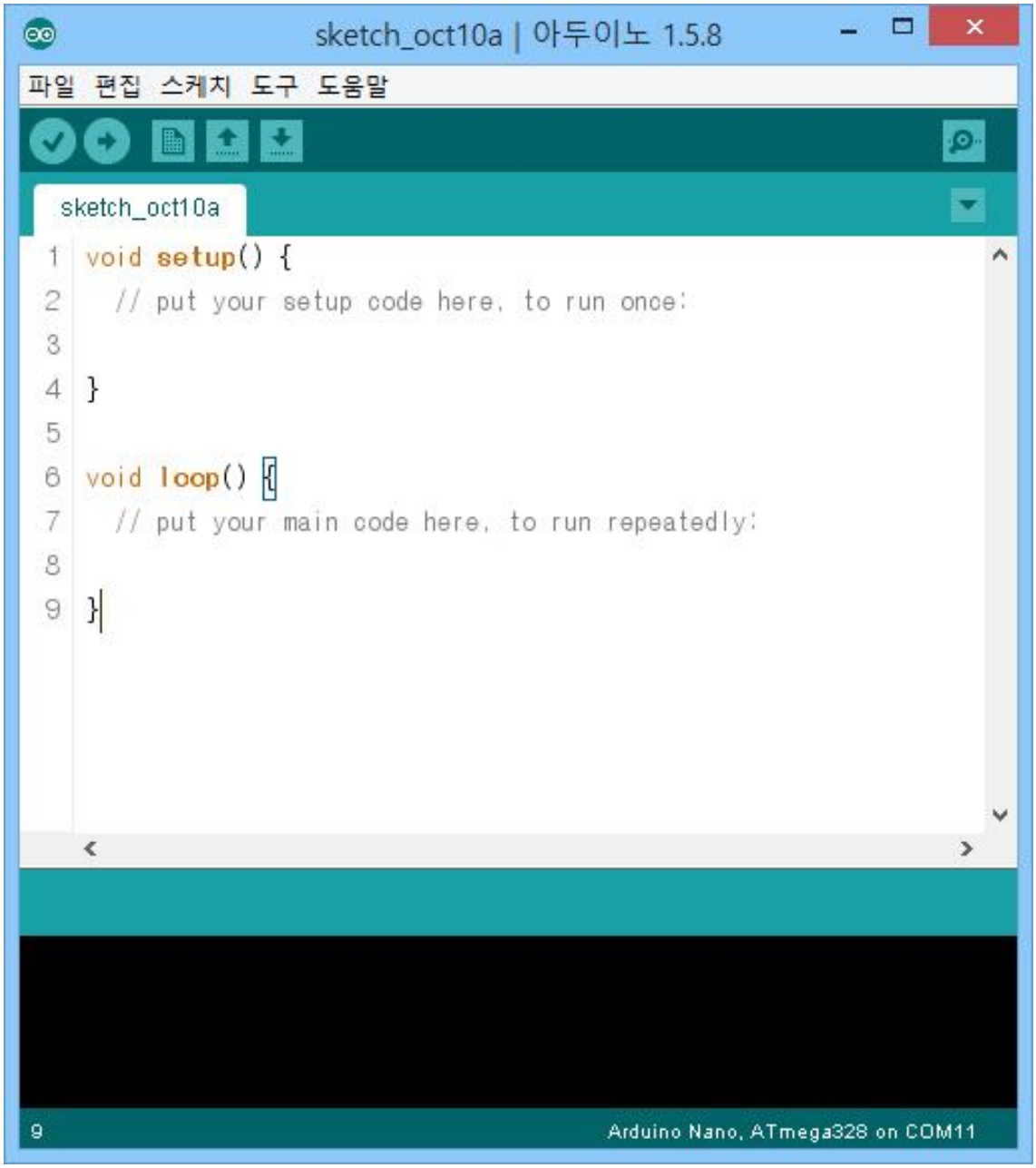
 열기

 저장

 상단 오른쪽에 시리얼 포트 모니터 창 열기 버튼이 있습니다.

2.4 스케치 프로그램 업로드

아래의 이미지처럼 기본 코드 상태에서 코드(code)를 기입합니다.
참고로 코드를 기입하는 행동을 코딩(coding) 이라고 합니다.
코딩(coding)을 하는 사람을 코더(coder)라고 합니다.
코더는 컴퓨터 프로그래머를 지칭하기도 합니다.
정식 명칭은 소프트웨어 엔지니어라고도 합니다.
또는 소프트웨어 개발자라고도 합니다.
스케치 IDE 를 사용하여 프로그래밍 하는 경우라면 펌웨어 소프트웨어 엔지니어라는 명칭이 어울립니다.
아두이노에서의 용어는 해외 개발자 사이트 등의 포럼을 보면 스케치 한다고 합니다.
새로운 용어이기도 합니다.
세련된 개념의 용어이기도 합니다.
관심 있으신 분들은 찾아 보시기 바랍니다.



위 이미지처럼 기본 코드 상태에서 아래의 코드를 입력 해 봅니다.

1 초마다 Running... 이라는 문자열을 시리얼 포트로 출력해주는 코드입니다.

```
void setup()
{
  Serial.begin(9600);
}
```

```
void loop()
{
  Serial.println("Running...");
  delay(1000);
}
```

현재 스케치의 코드 컴파일->링크 과정을 거쳐서 나온 hex 파일이 아두이노에 자동으로 업로드 됩니다.

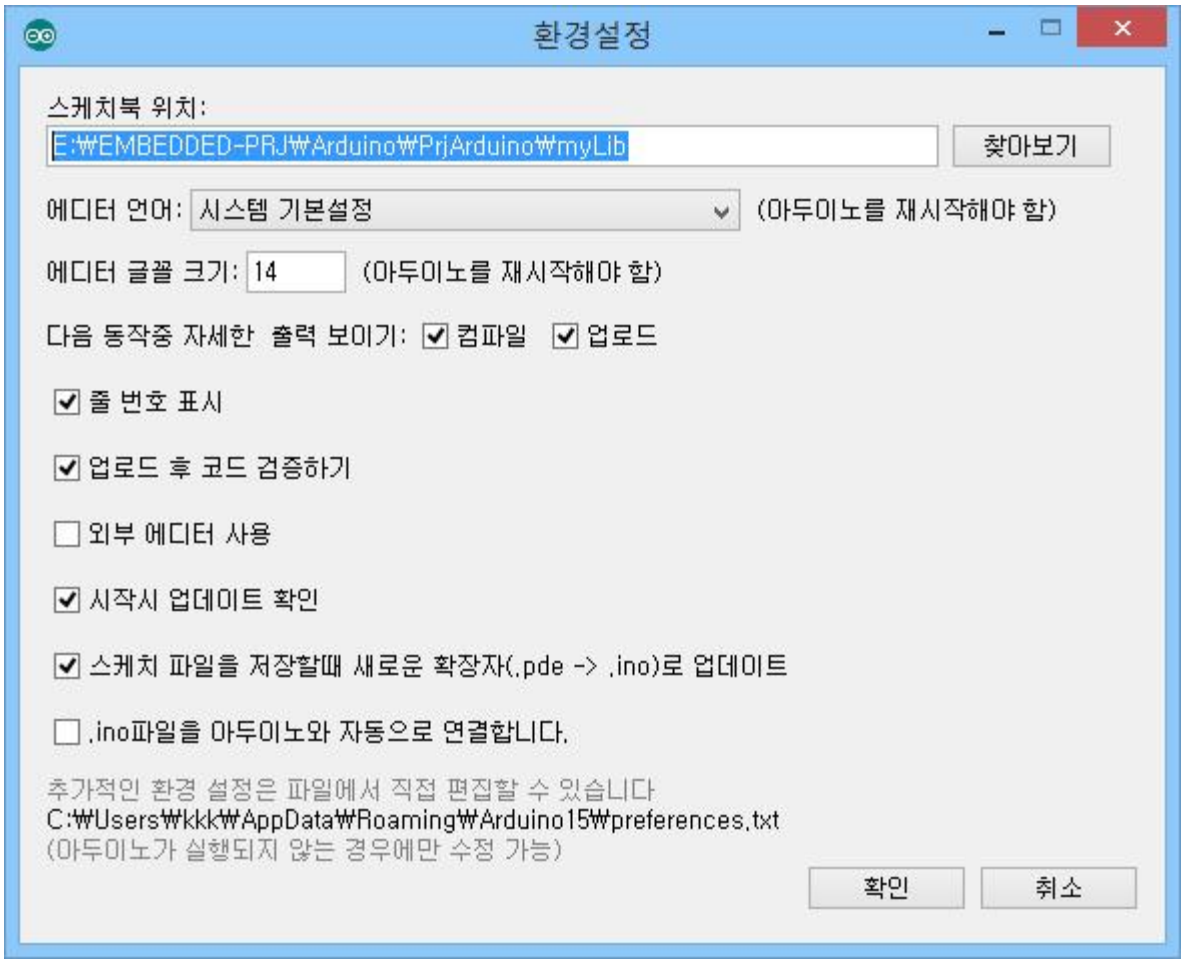
성공적인 업로드가 되었다면 1 초마다 "Running..." 이라는 문자열을 시리얼 포트로 출력하는 코드입니다.
시리얼 모니터 창을 열어서 출력되는 문자열을 볼 수 있습니다.
시리얼 모니터 창은 스케치 IDE 의 메뉴 도구 -> 시리얼 모니터 선택하면 나옵니다.

3 스케치 IDE 환경 설정

4 스케치 IDE 에 라이브러리 추가하기

4.1 라이브러리 디렉터리 확인 방법

스케치 IDE 메뉴->파일->"환경 설정" 선택하면 아래의 다이얼로그 보입니다.



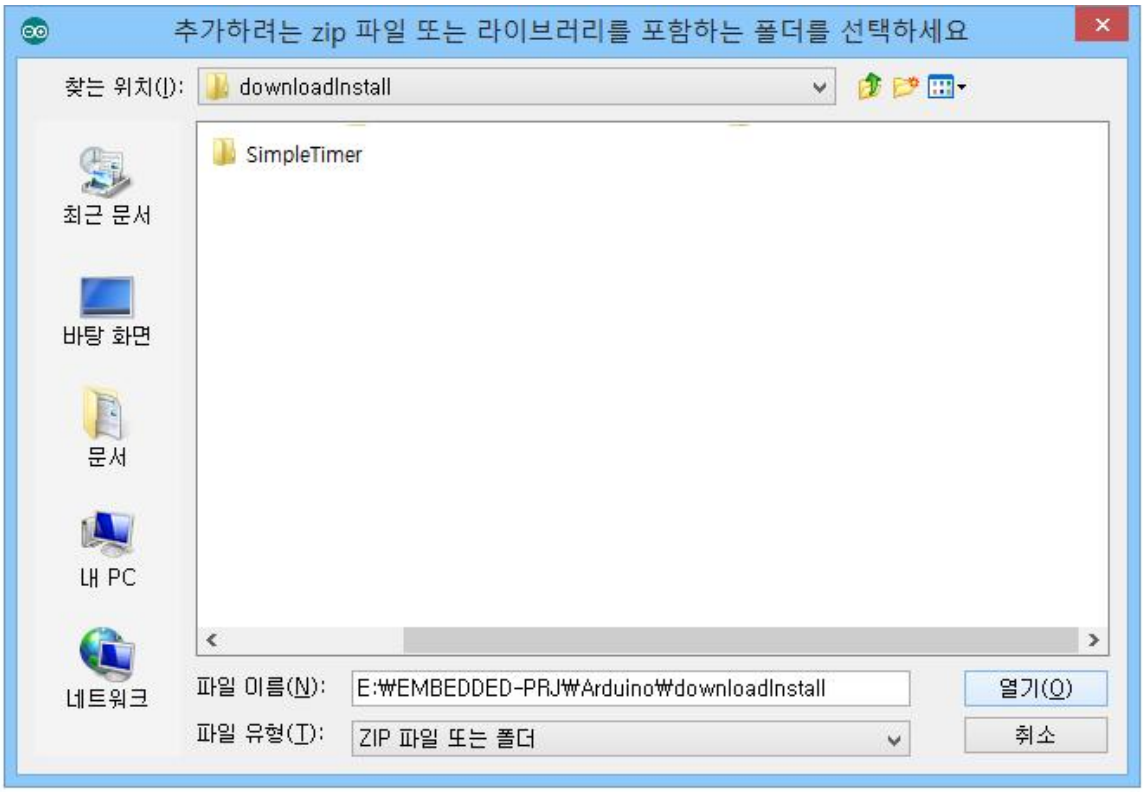
"스케치북 위치:" 기본 위치는 윈도우 사용자 My Documents 아래의 디렉터리로 설정되어 있습니다.
기본 설정대로 사용, 또는 임의의 원하는 위치의 디렉터리를 지정하여 사용 할 수도 있습니다.

4.2 라이브러리 추가 방법 1

스케치 IDE 에서의 "라이브러리 추가" 메뉴를 이용할 수 있습니다.

스케치 IDE 메뉴 → 스케치 → 라이브러리 가져오기 → (서브 메뉴 상단)"라이브러리 추가"

파일 선택 다이얼로그가 나옵니다.



라이브러리 파일 ZIP (압축된 파일 형식) 또는 폴더를 지정할 수 있습니다.

ZIP 파일 또는 폴더를 선택하면 Zip 파일일 경우 압축 해제되어 라이브러리 디렉터리(폴더)에 복사 됩니다.

4.3 라이브러리 추가 방법 2

환경 설정에 정의된 라이브러리 디렉터리 아래에 직접 파일들을 넣어줍니다.
스케치 라이브러리 디렉터리가 "E:₩EMBEDDED-PRJ₩Arduino₩PrjArduino₩myLib" 라고 설정된 상태일 경우 윈도우 탐색기에서 해당 디렉터리에 파일을 넣어주면 됩니다.
E:₩EMBEDDED-PRJ₩Arduino₩PrjArduino₩myLib 디렉터리일 경우 하위 디렉터리 "libraries" 하위 디렉토리로 복사해 줍니다.

스케치 IDE 예제 업로드 & 테스트

스케치 IDE 프로그램에는 많은 기본 예제들이 있습니다.
LED 깜박이는 예제를 빌드 & 업로드 & 테스트 해봅니다.
스케치 IDE 메뉴 -> 파일 -> 예제 -> 많은 라이브러리 항목이 보입니다.

감사합니다

http://www.irnumall.co.kr