아두이노 피지컬컴퓨팅

2024. C-MOOC DAY

충북교육연구정보원 강사 박정진

- 강의자료: https://arduino.datahub.pe.kr/
- 프로그램: https://arduino.datahub.pe.kr/sw
- 소스코드: https://arduino.datahub.pe.kr/src

피지컬 컴퓨팅

- 의미
  - **현실 세계의** 아날로그 **정보를 인지하여 그에 맞게 대응할 수 있도록** 센서와 마이크로컨트롤러 등의 **하드웨어 장치와 소프트웨어로 컴퓨팅 시스템을 만드는 것**.
- 기원
  - 뉴 미디어 아트(new media art) 예술가들이 컴퓨터를 이해하여 영상물이나 빛을 이용한 작품을 만들기 위해서 사용하고 발전시킴
- 활용
  - **창의적 컴퓨팅과 소프트웨어 교육에 많이 활용**된다. 학생들은 물리적인 것을 실제로 보고 만지고 느끼면서 **재미**를 느끼고 컴퓨터, 알고리즘 등에 대해 쉽게 배울 수 있다.

에스컬레이터

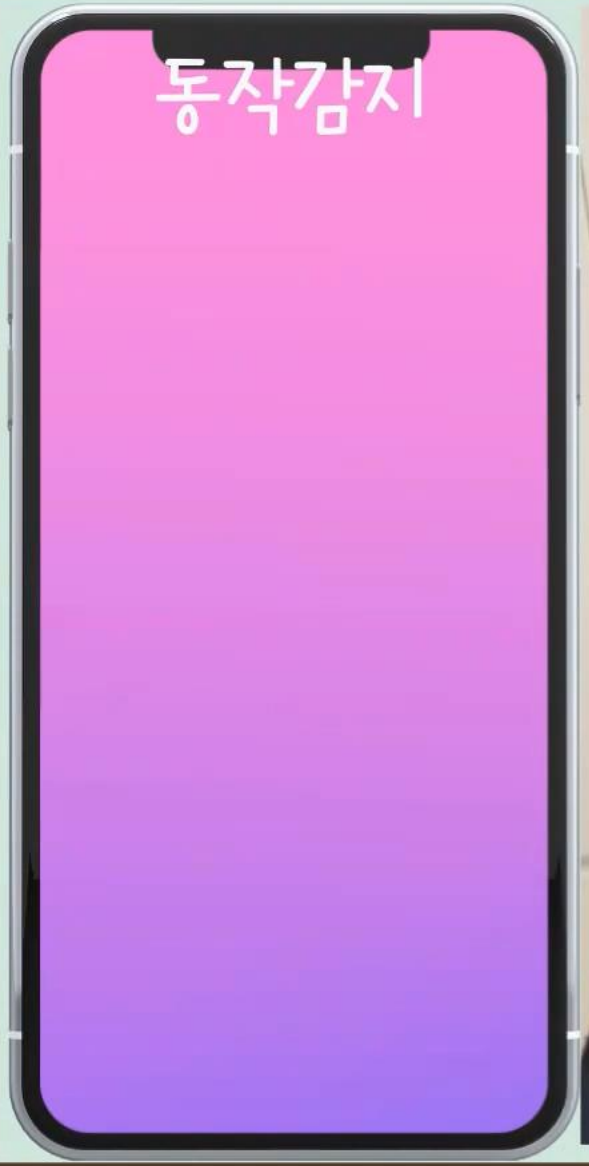
자동문

자동문

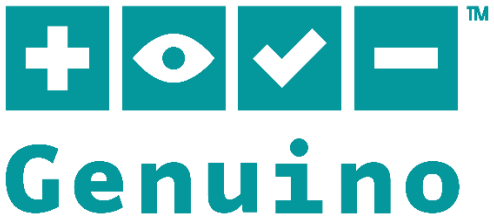
자동 센서 수도꼭지

계단 센서등

동작감지



지문인식



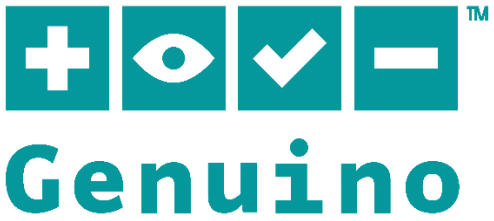
- **아두이노(arduino)는** AVR이라는 마이크로 컨트롤러를 기반으로 한 오픈소스 **원보드 마이컴**이다. 예전에는 이런 시스템을 설계하고 다루기 위해서는 전기/전자와 관련된 전문적인 기술이나 지식이 필요했다.
- 이 보드는 그러한 어려움을 극복하고자 **기술 숙련도가 낮은 디자이너 혹은 예술가들을 대상으로 설계된 것**이다. **따라서 비전공자들도 손쉽게 익히고 사용할 수 있다는 큰 장점**을 갖는다.
- 그러면 이러한 아두이노 보드를 이용하여 어떤 일들을 할 수 있을까.
  - 사람이 접근하면 자동으로 조명이 켜지는 장치.
  - 애완 동물에게 정해진 시간에 먹이를 공급하는 장치.
  - 화분의 습도가 낮아지면 자동으로 물을 주라는 트윗을 보내는 장치.
  - 여러개의 LED를 이용하여 귀걸이나 목걸이의 장식으로 이용.

ARDUINO™ Genuino™

- Arduino 의 구성

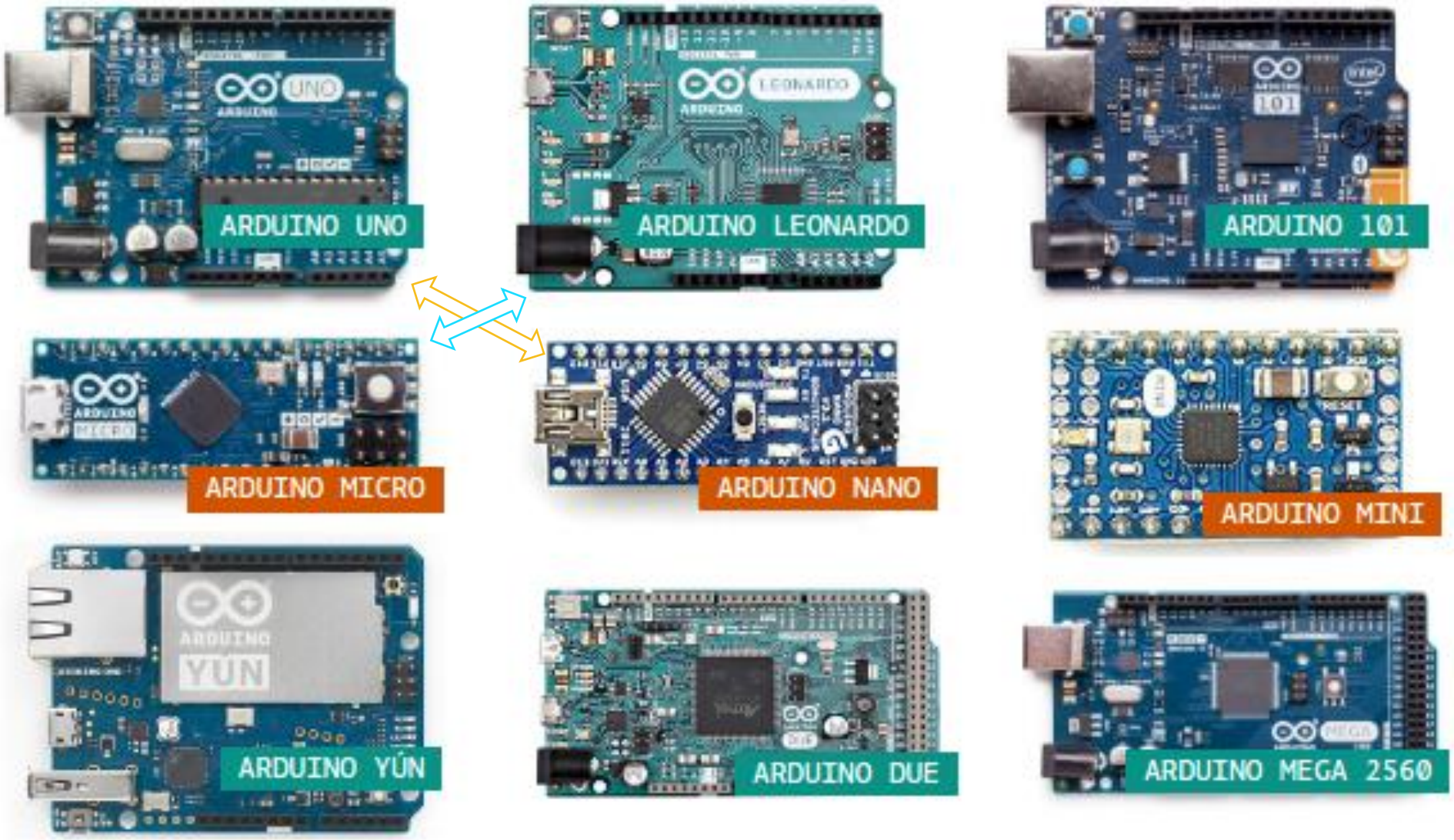
Arduino 아두이노 = H/W (아두이노 보드) + S/W (아두이노 소프트웨어) + 오픈 소스

- 공식사이트
  - http://arduino.cc/



- 특장점
  - 1. 저비용 : 아두이노 보드는 다른 마이크로컨트롤러 플랫폼에 비해 저렴하다.
  - 2. 크로스 플랫폼 : 아두이노 소프트웨어는 윈도우즈, 맥OSX, 리눅스 운영체제 모두에서 작동한다.
  - 3. 간단하고 명확한 프로그래밍 환경 : 아두이노 프로그래밍 환경은 초보자들이 사용하기 쉬울 뿐 아니라 실력자들이 여러가지 다양한 시도를 하기 위한 유연성을 제공한다. 소프트웨어 개발을 위한 통합개발환경(IDE)가 제공되며 컴파일된 펌웨어(특정 하드웨어 상에서 동작하는 소프트웨어)를 USB를 통해 손쉽게 업로드 할 수 있다.
  - 4. 오픈 소스 : 아두이노 하드웨어 및 소프트웨어는 오픈 소스 툴이기 때문에 고급 프로그래머들에 의해 작성된 확장 소프트웨어 라이브러리들을 구할 수 있으며, 회로 설계자들이 손쉽게 자신만의 모듈을 만들고 개선할 수 있다.

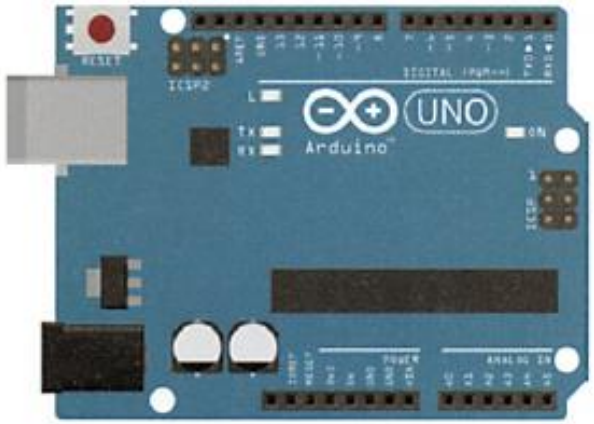
아두이노 시리즈


ARDUINO UNO
ARDUINO LEONARDO
ARDUINO 101
ARDUINO MICRO
ARDUINO NANO
ARDUINO MINI
ARDUINO YÚN
ARDUINO DUE
ARDUINO MEGA 2560

아두이노 종류

- Arduino Uno

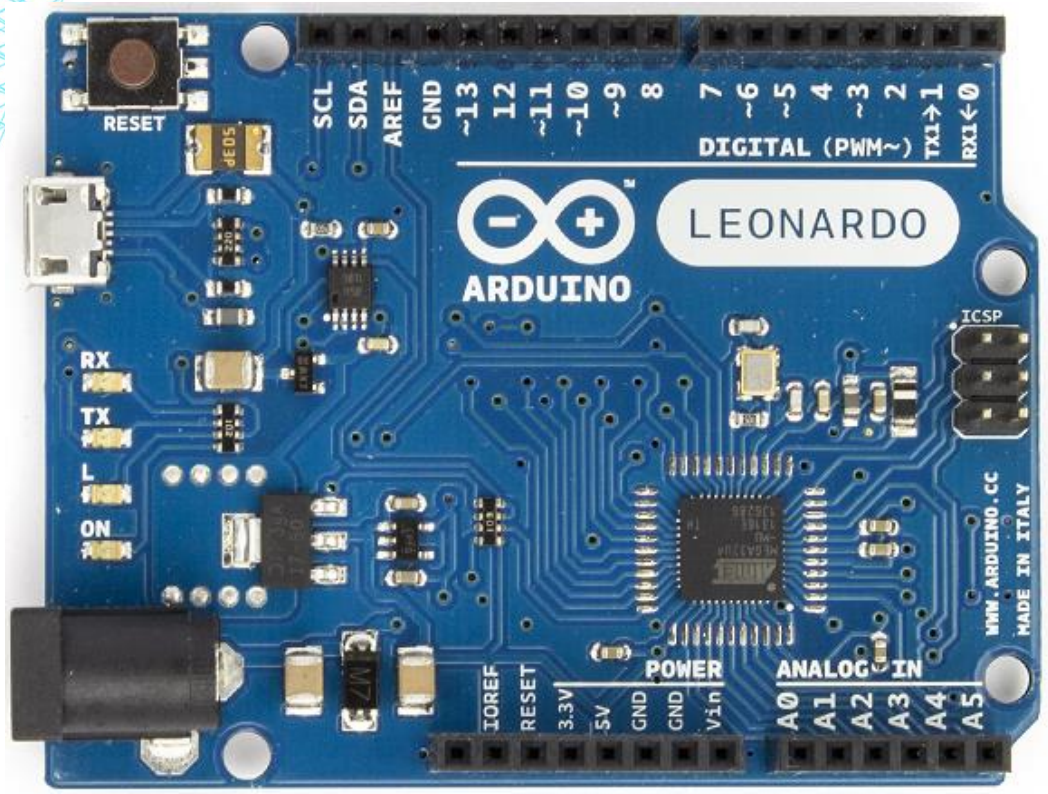
아두이노 우노는 영어로 Arduino UNO라고 씁니다. UNO는 이탈리아어로 하나를 뜻하는 것으로 아두이노社의 첫 번째 아두이노라는 것을 나타내는 말입니다. 사람들에게 가장 많이 사랑받는 기종으로 크기가 5.3cm × 7.7cm 정도로 작은 기종입니다. 여러 개의 구멍이 있어 핀을 꽂아서 사용하도록 만든 표준모델입니다. 아날로그 6개, 디지털 14개로 총 20개의 연결단자를 제공합니다.



개발에 사용하는 컴퓨터를 'PC'라고 부릅니다. PC와 연결은 B type USB 케이블을 연결합니다. 전원공급단자를 제외하고 별도의 전원을 연결해서 사용할 수 있도록 되어 있습니다. 별도의 전원은 7V에서 12V 전원을 연결할 수 있습니다.

아두이노 종류

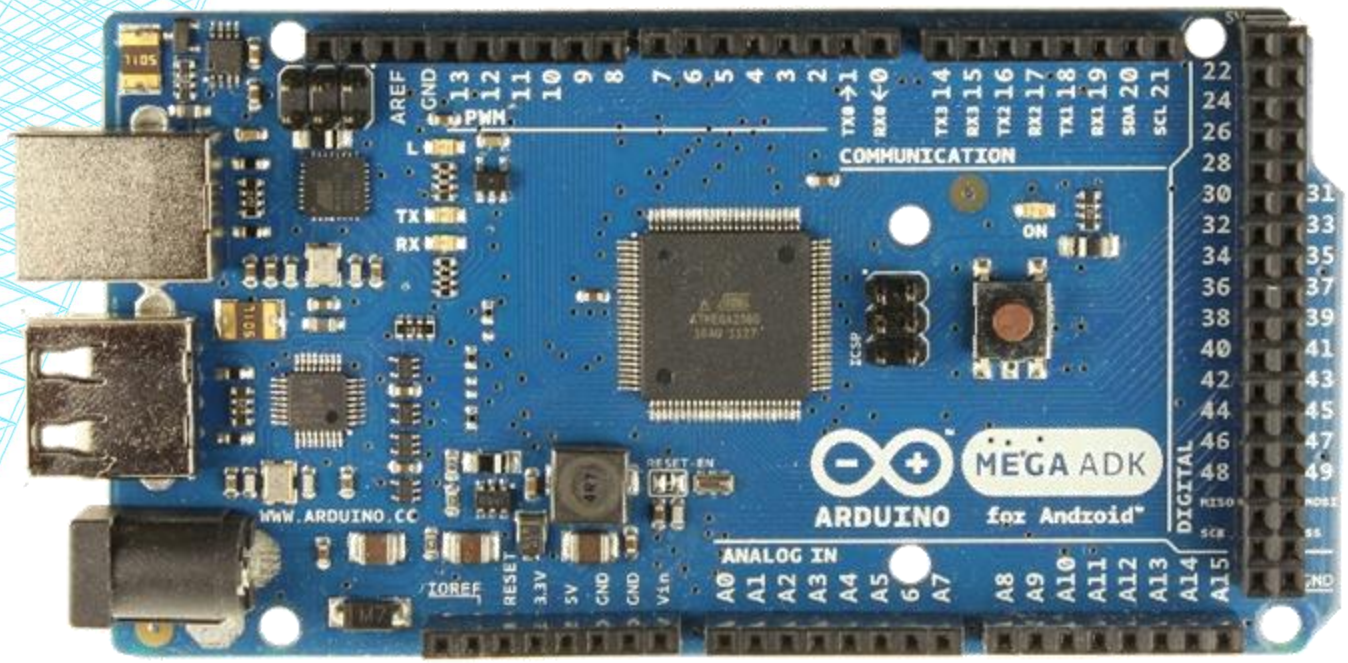
- Arduino Leonardo



- USB 인터페이스가 내장되어 USB 장치 에뮬레이팅이 가능한 보드
- 레오나르도 보드에서는 PC와의 통신을 Serial, D0-D1 핀을 이용한 serial 통신을 Serial1 으로 사용할 수 있도록 지정되어 있습니다. 별도의 serial 통신용 핀이 더 생긴 셈이기 때문에 고속으로 동작하는 통신 모듈을 함께 사용할 때 유리
- 특수한 기능(PWM, I2C, SPI 등)을 담당하는 핀이 아두이노 UNO와는 완전히 틀리기 때문에 UNO 용 라이브러리들이 동작하지 않을 수 있음.
- 아두이노 우노는 USB 통신용 칩이 따로 붙어있어 총 두 개의 칩이 있는데, 아두이노 레오나르도는 하나의 칩으로 통합하여 사용됨
- 컴퓨터에 연결하면 컴퓨터가 주변 입력장치(HID)로 인식하여, 키보드나, 마우스, 조이스틱과 같이 취급합니다. 즉, 키보드 대용으로 추가적인 키패드를 만들거나 특정 매크로를 실행하는 디바이스 제작에 활용 가능

아두이노 종류

- Arduino Mega



- UNO 보드의 확장 버전
- UNO 보드의 약 2배 크기
- 많은 IO 핀을 가지고 있고, 더 빠르고, 저장용량도 더 많이 가지고 있어 다른 보드에 비해 훨씬 많은 기가와 통신할 수 있는 고성능 보드
- UNO 보드로 처리하기 힘든 멀티미디어 관련 작업, 로봇이나 이미지, 음성, 영상 등과 같은 복잡한 제어가 필요한 작업에 사용되어 짐

아두이노 대체 보드

- ESP8266
  - WIFI 기능추가

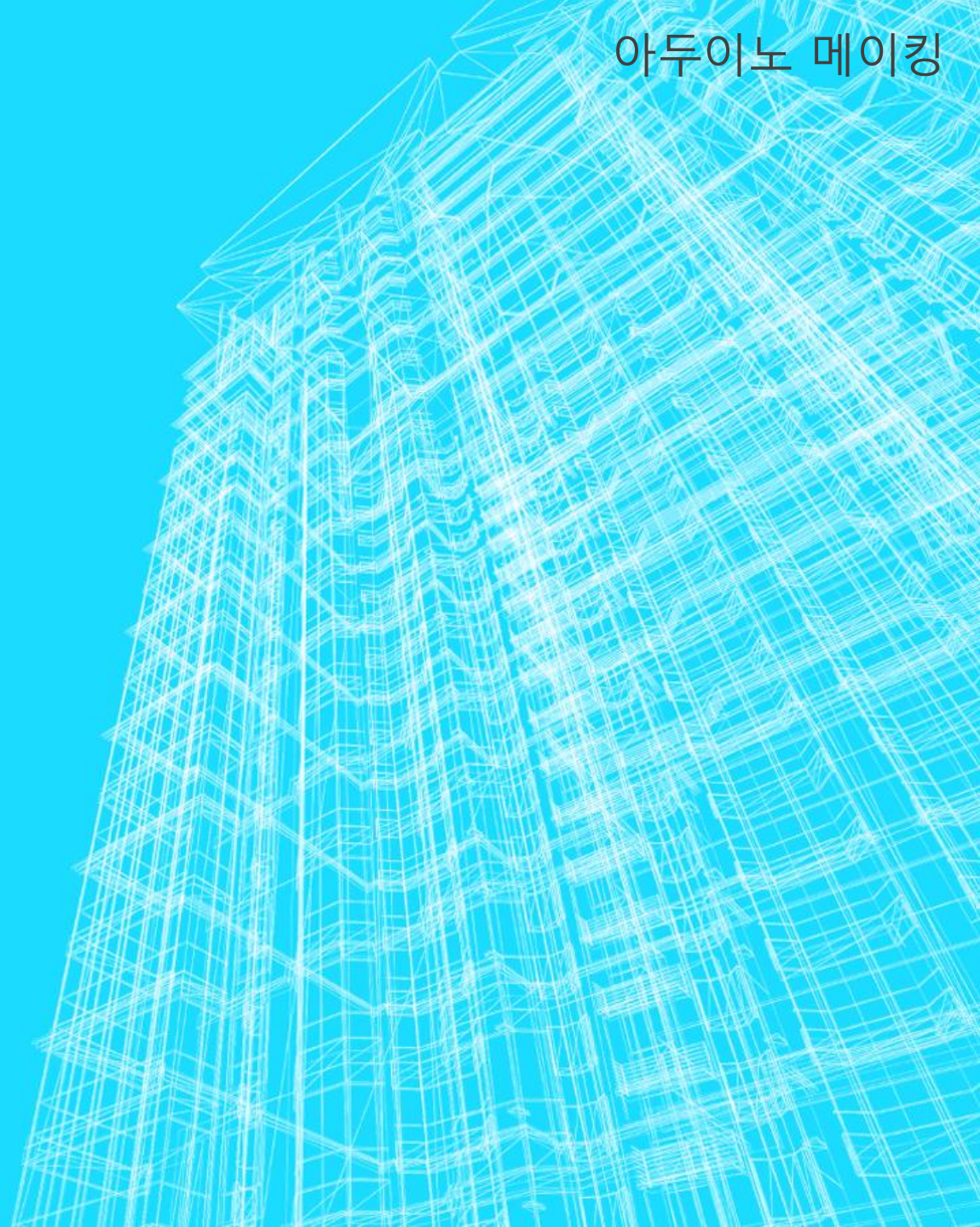
- ESP32
  - WiFi + Bluetooth 기능 추가

Arduino UNO vs ESP8266 vs ESP32

	Arduino UNO	ESP8266	ESP32
MCU	ATmega328P 8bit	Xtensa Single-Code 32bit L106	Xtensa Dual-Core 32bit LX6
Frequency	16Mhz	80Mhz	160Mhz
802.11 b/g/n Wi-Fi	None	Yes, HT20	Yes, HT40
Bluetooth	None	None	Bluetooth 4.2 and BLE
SRAM	2KB	160kB	512kB
FLASH	32KB	16MB	16MB
GPIO	14	17	36
HW / SW PWM	5/0	0/8	1/16
SPI/I2C/I2S/UART/CAN	1/1/0/1/1	2/1/2/2/0	4/2/2/2/1
ADC pins	6 (10-bit)	1 (10-bit)	18 (12bit)
DAC pins	0	0	3
Touch Sensor	None	None	Yes
Temperature Sensor	None	None	Yes

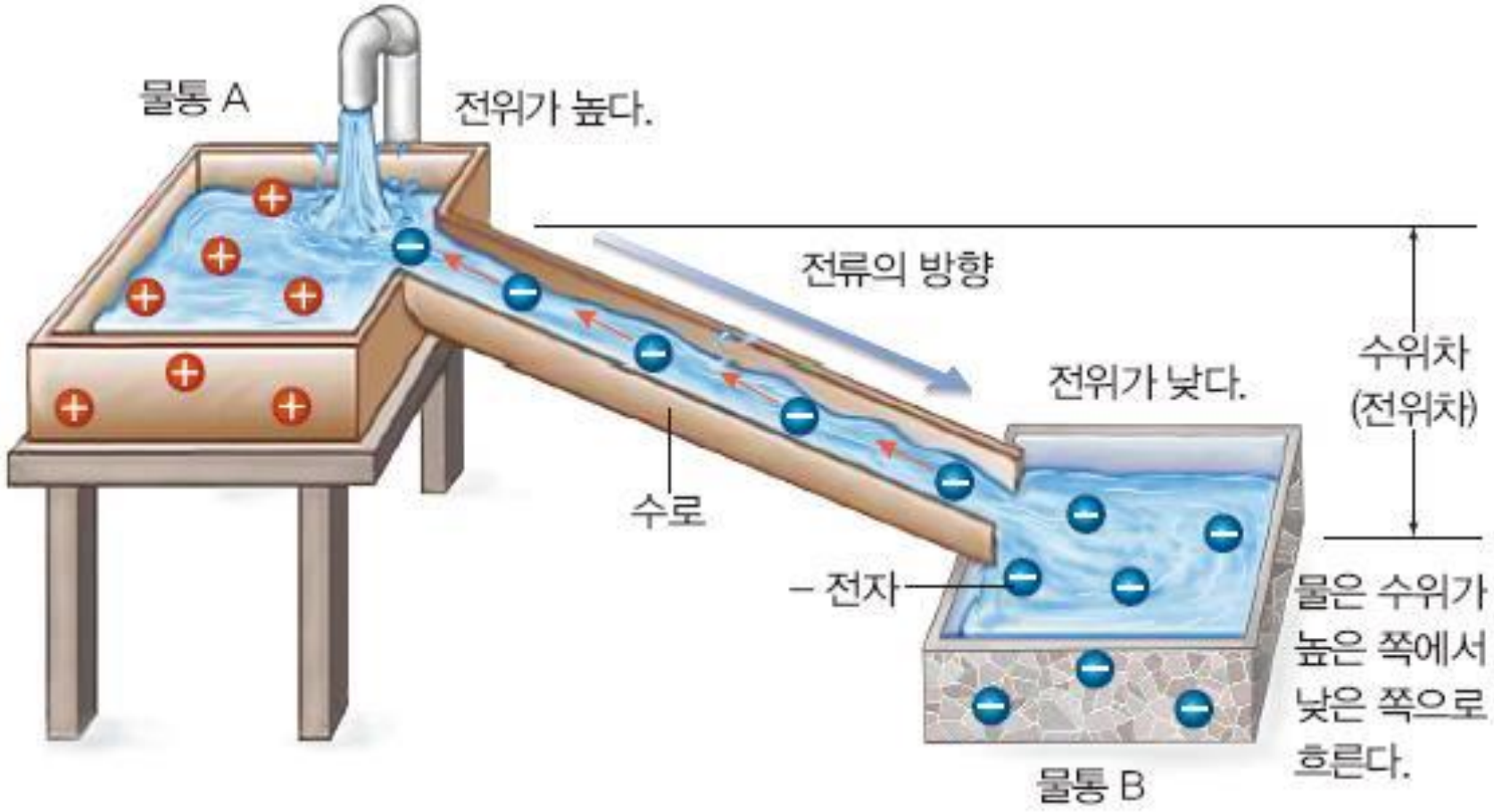
준비운동

피지컬컴퓨팅 강사 박정진

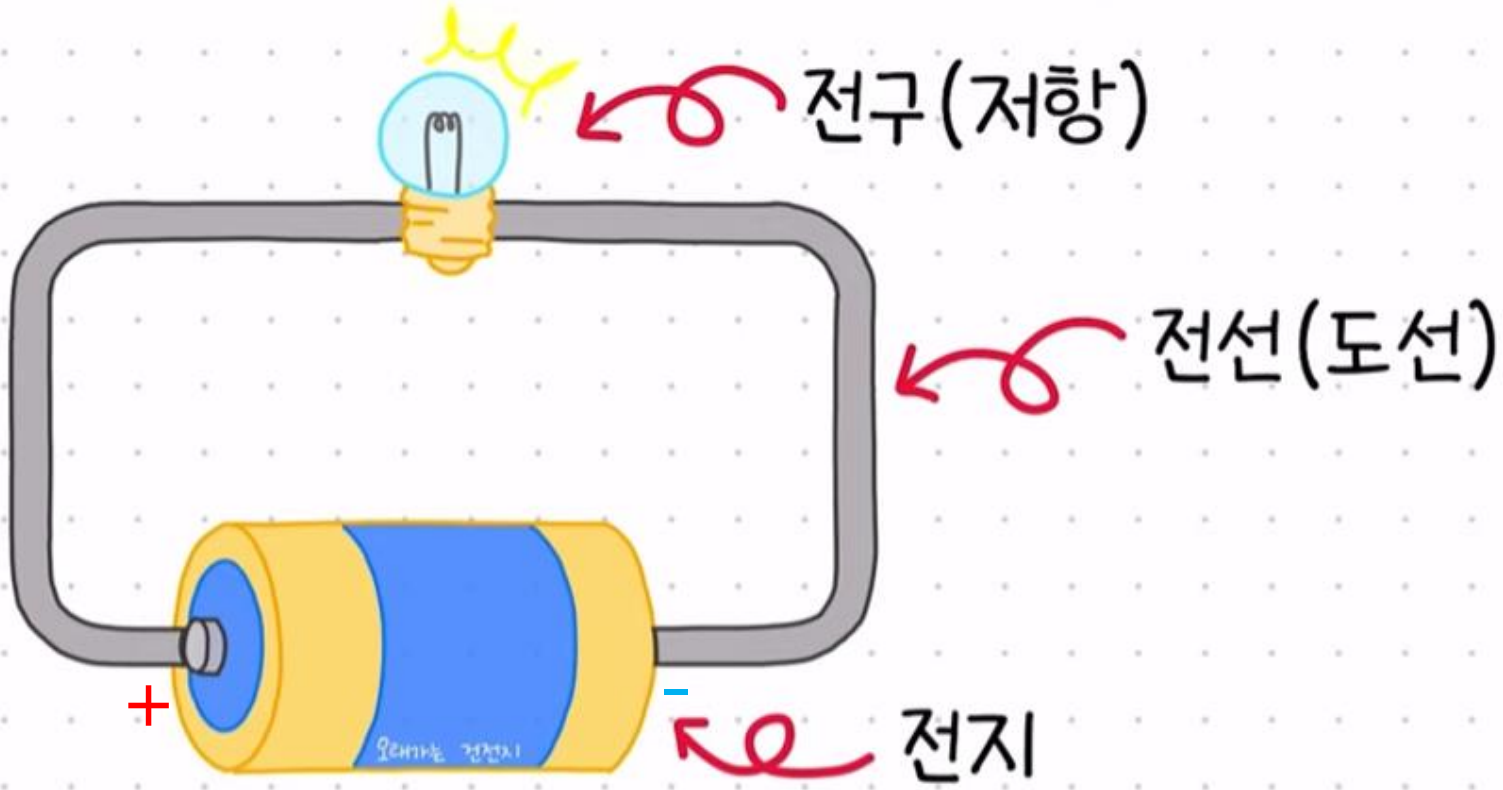


전압과 전류

- 전압: 전기의 위치 에너지 차이



복습하기



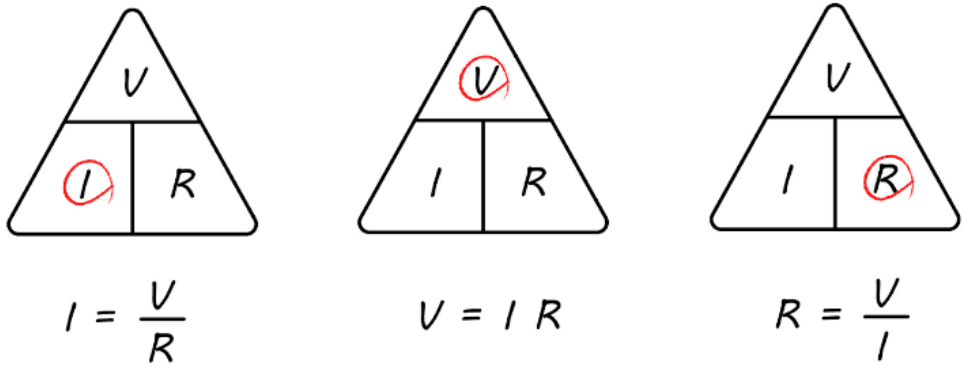
구분	정의	단위
전류	전하의 흐름(전자의 이동 속도)	암페어(A)
전압	전류를 흐르게 하는 전지의 능력	볼트(V)
저항	전류의 흐름을 방해하는 것	옴(Ω)

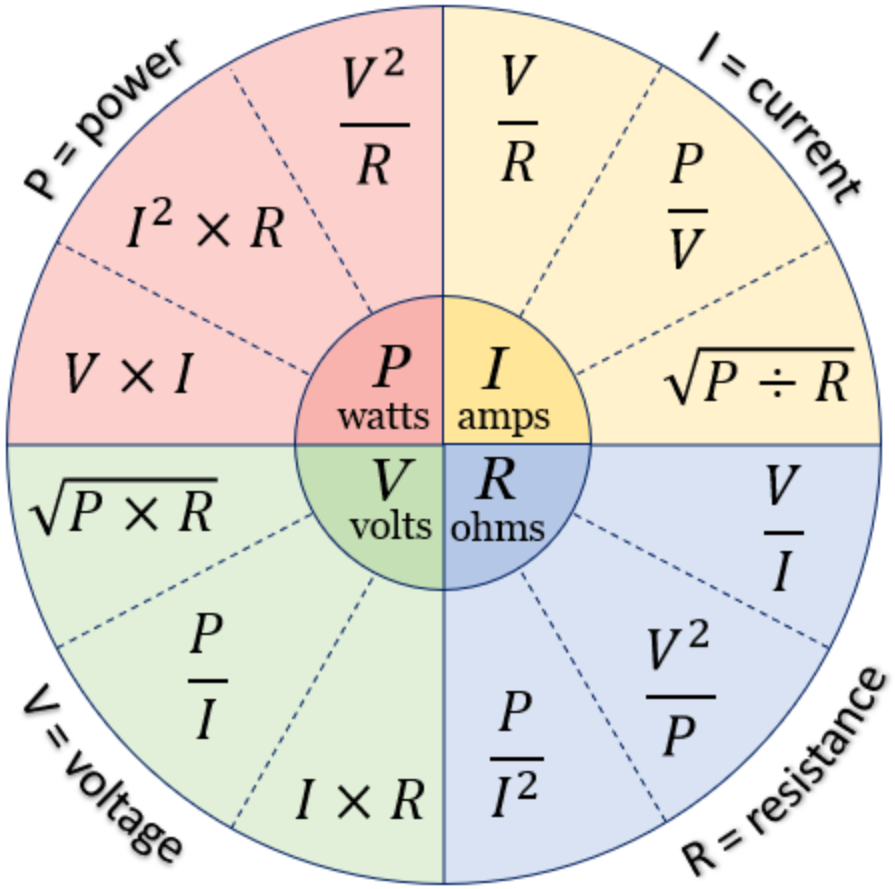
옴의 법칙

- 표기 / 단위

Quantity	Symbol	Unit of Measurement	Unit Abbreviation
Current	I	Ampere ("Amp")	A
Voltage	E *or* V	Volt	V
Resistance	R	Ohm	Ω

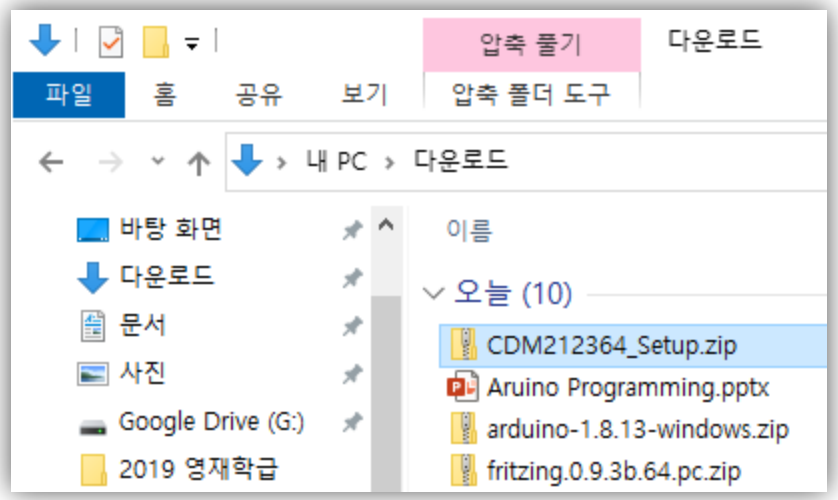
- 옴의 법칙

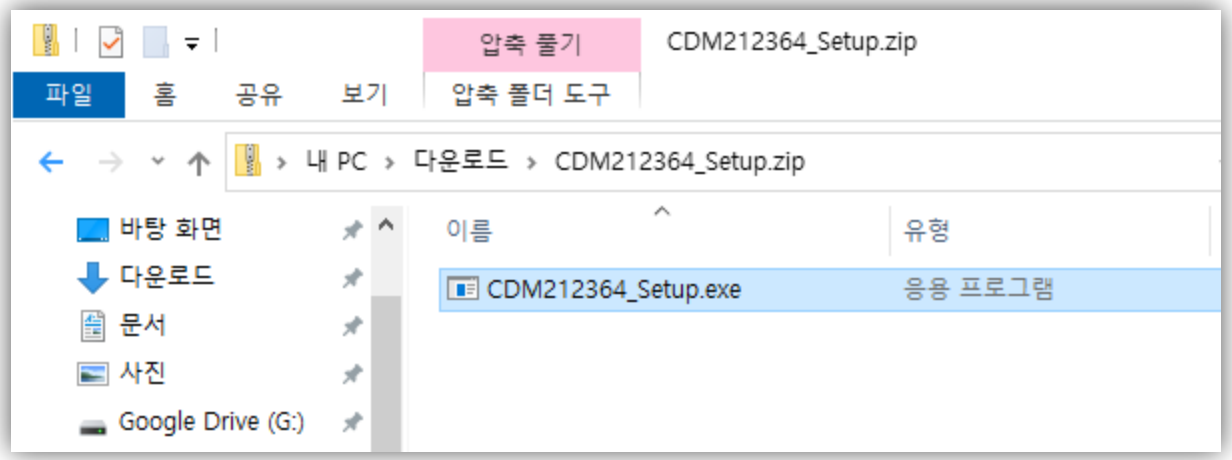




아두이노 호환보드 설정(FTDI)

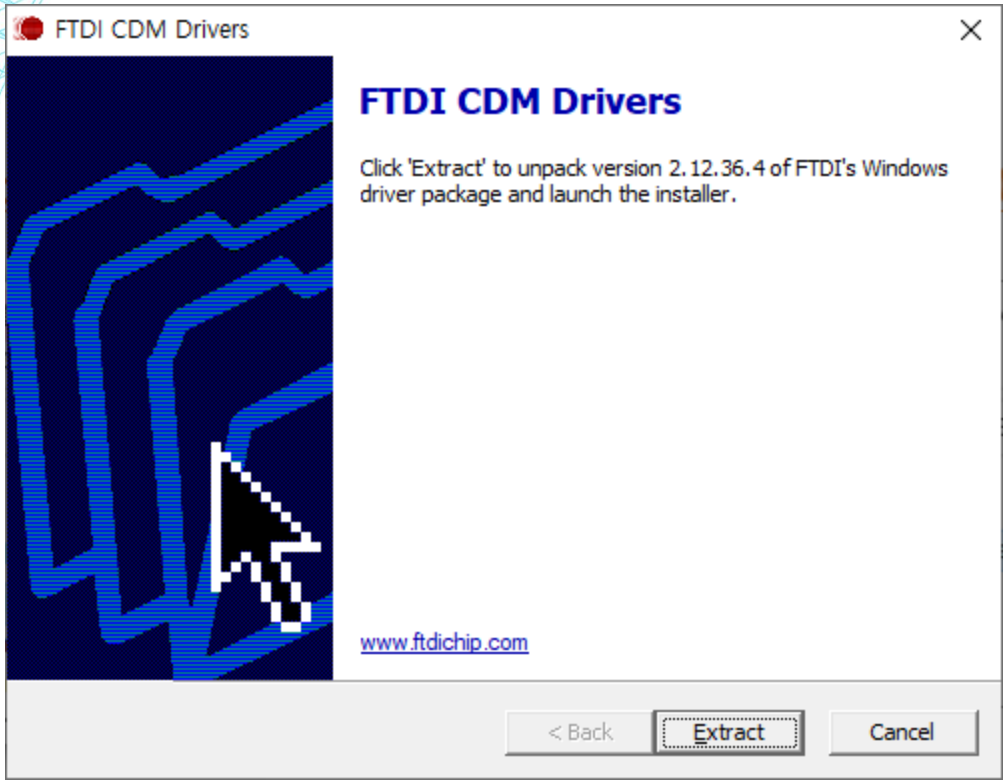
- FTDI 드라이버 설치
  - https://ftdichip.com/wp-content/uploads/2021/08/CDM212364_Setup.zip

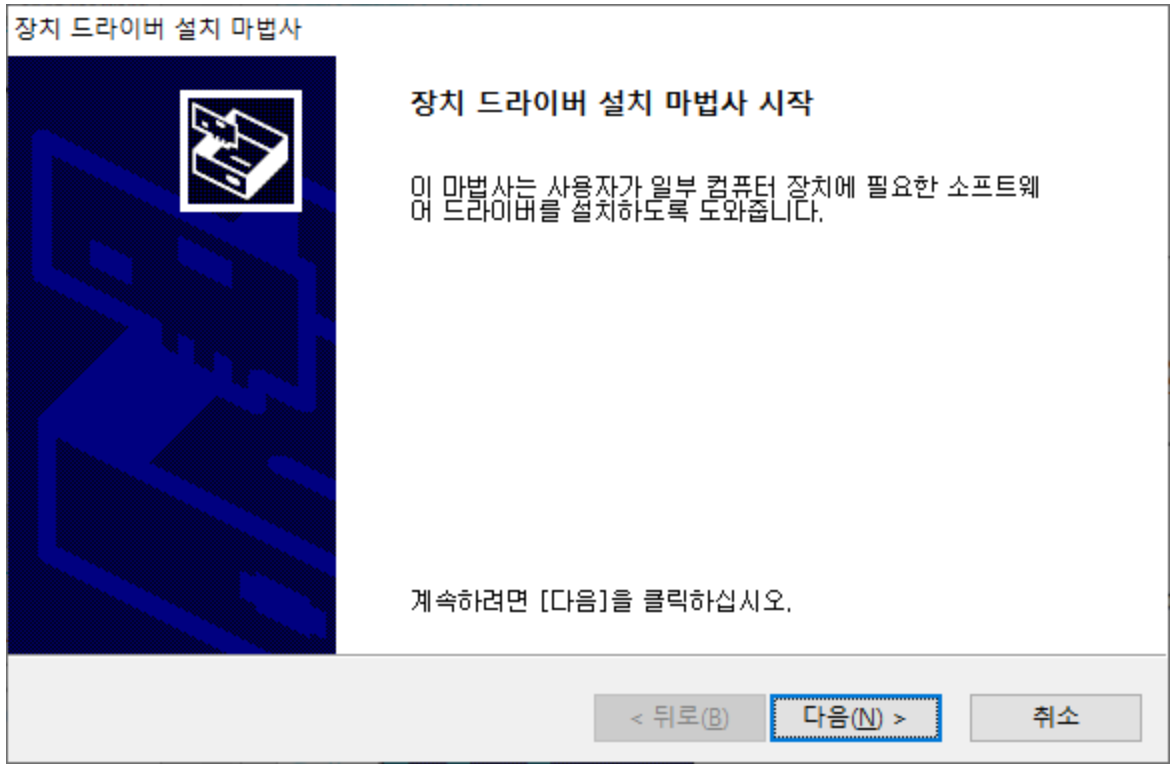




아두이노 호환보드 설정(FTDI)

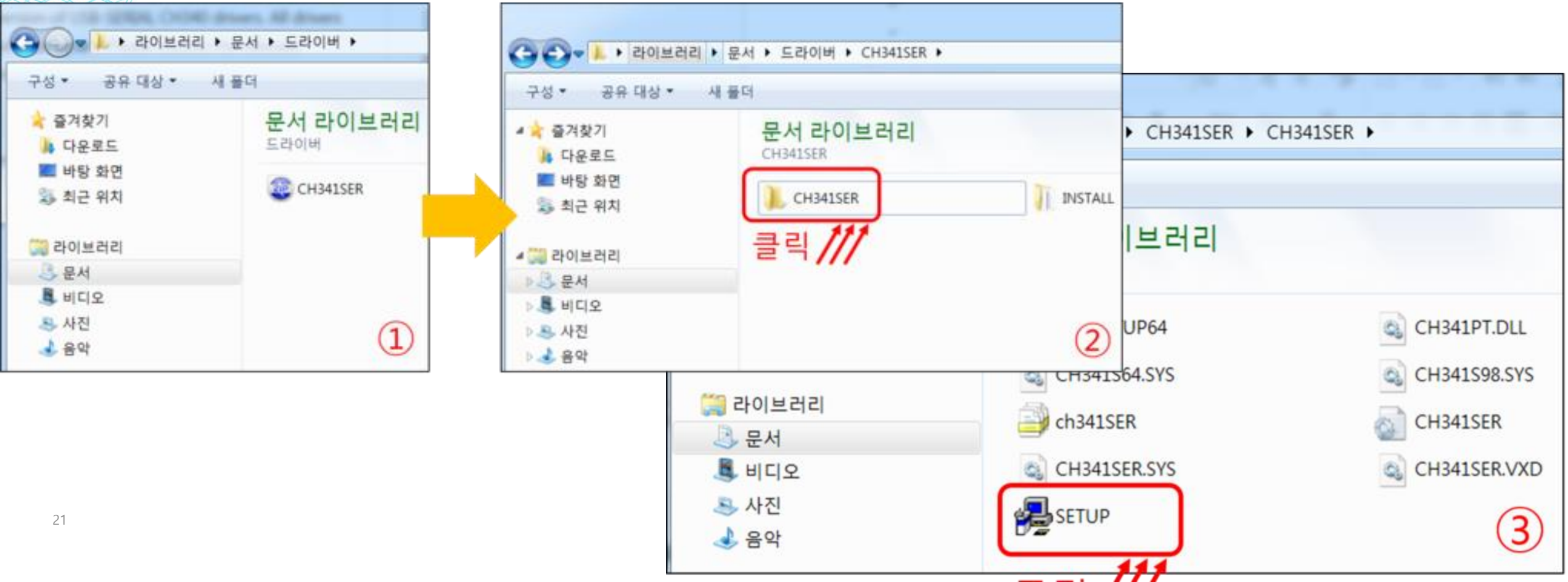
- FTDI 드라이버 설치
  - https://ftdichip.com/wp-content/uploads/2021/08/CDM212364_Setup.zip





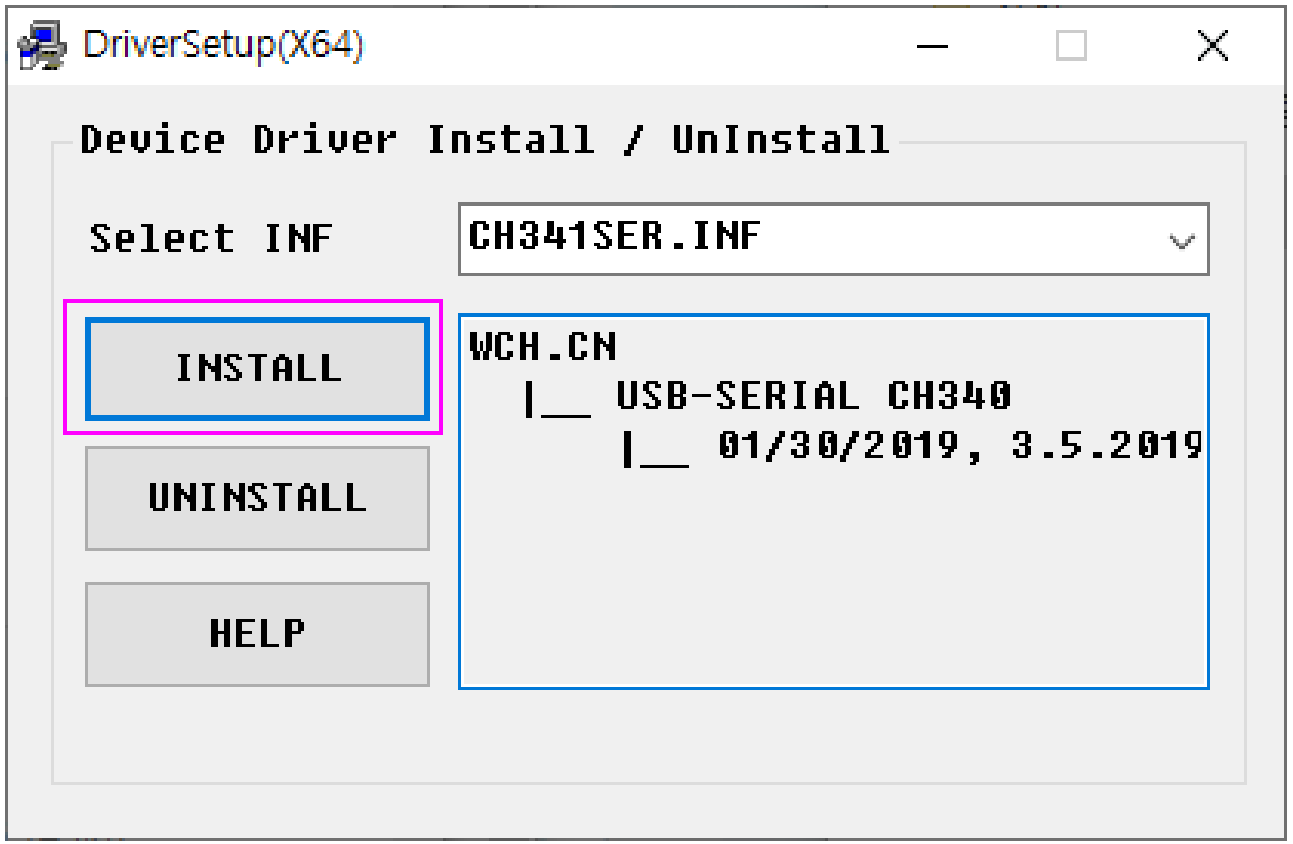
아두이노 호환보드 설정(CH340)

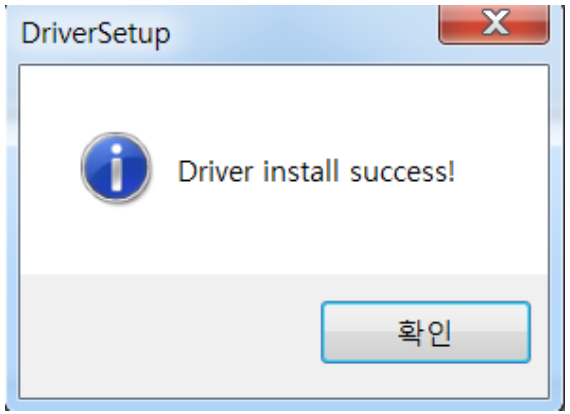
- CH341 드라이버 설치
  - 다운로드 링크: http://arduino.datahub.pe.kr/sw/CH341SER%20Driver.zip



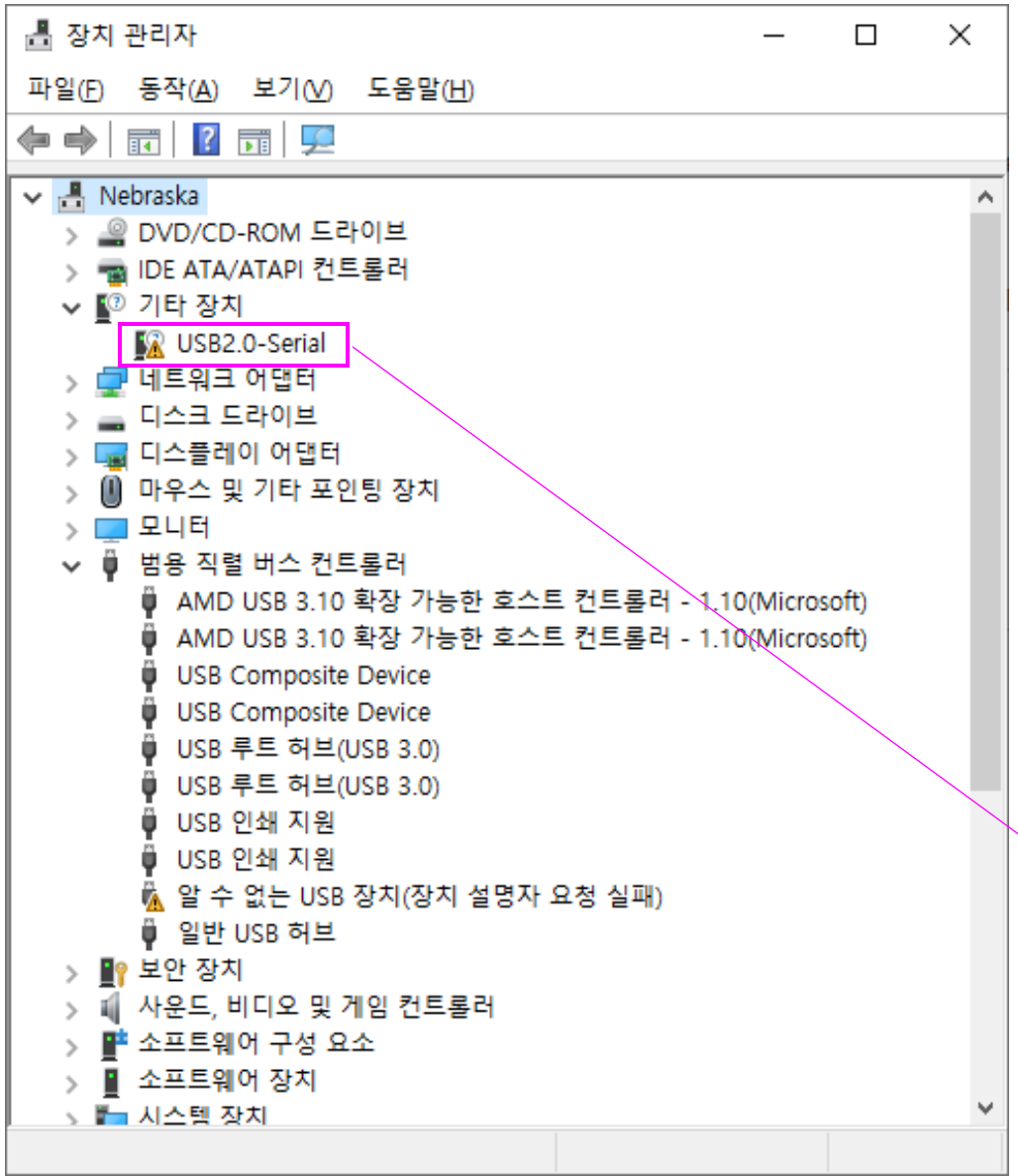
아두이노 호환보드 설정(CH340)

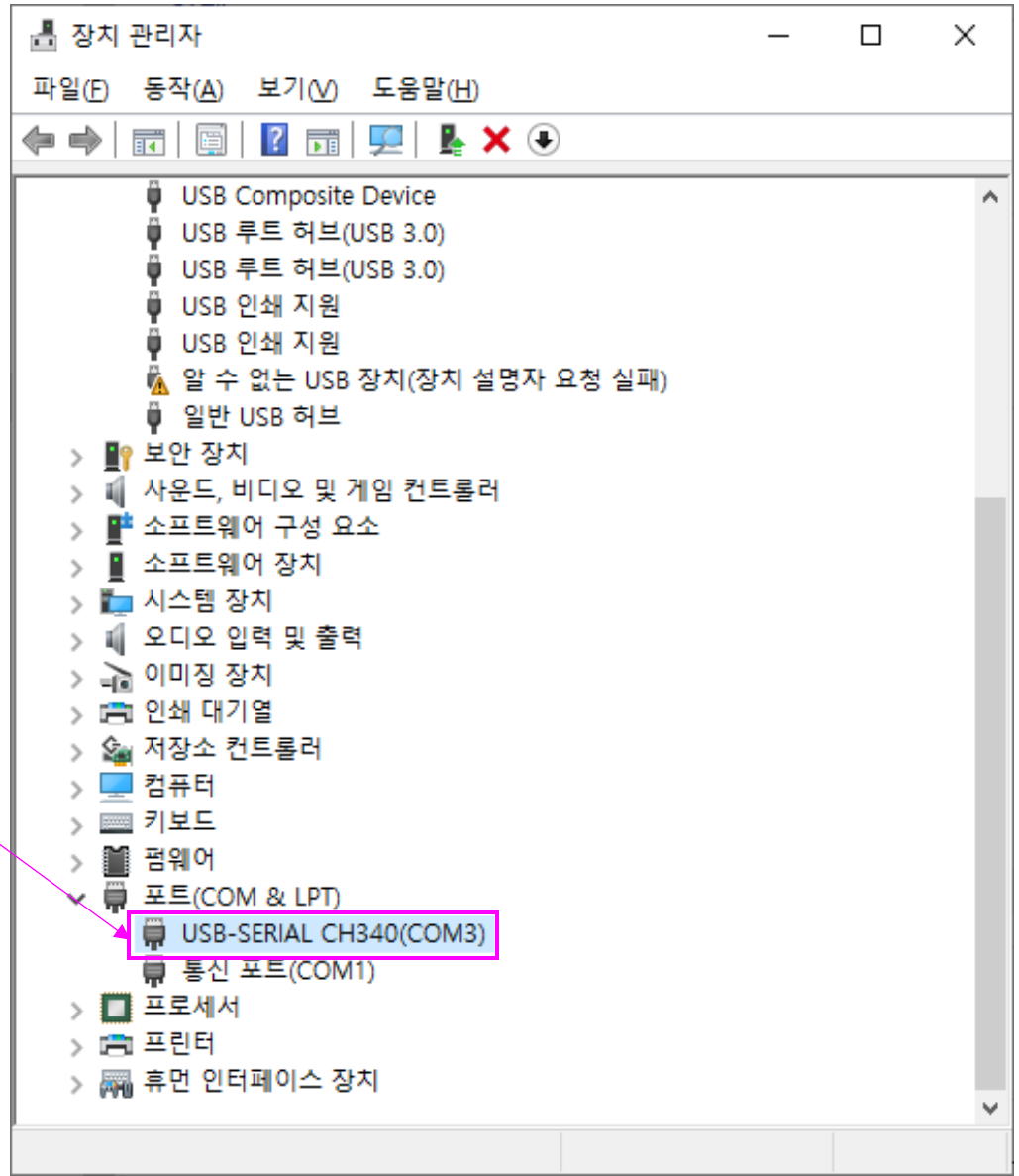
- CH341 드라이버 설치



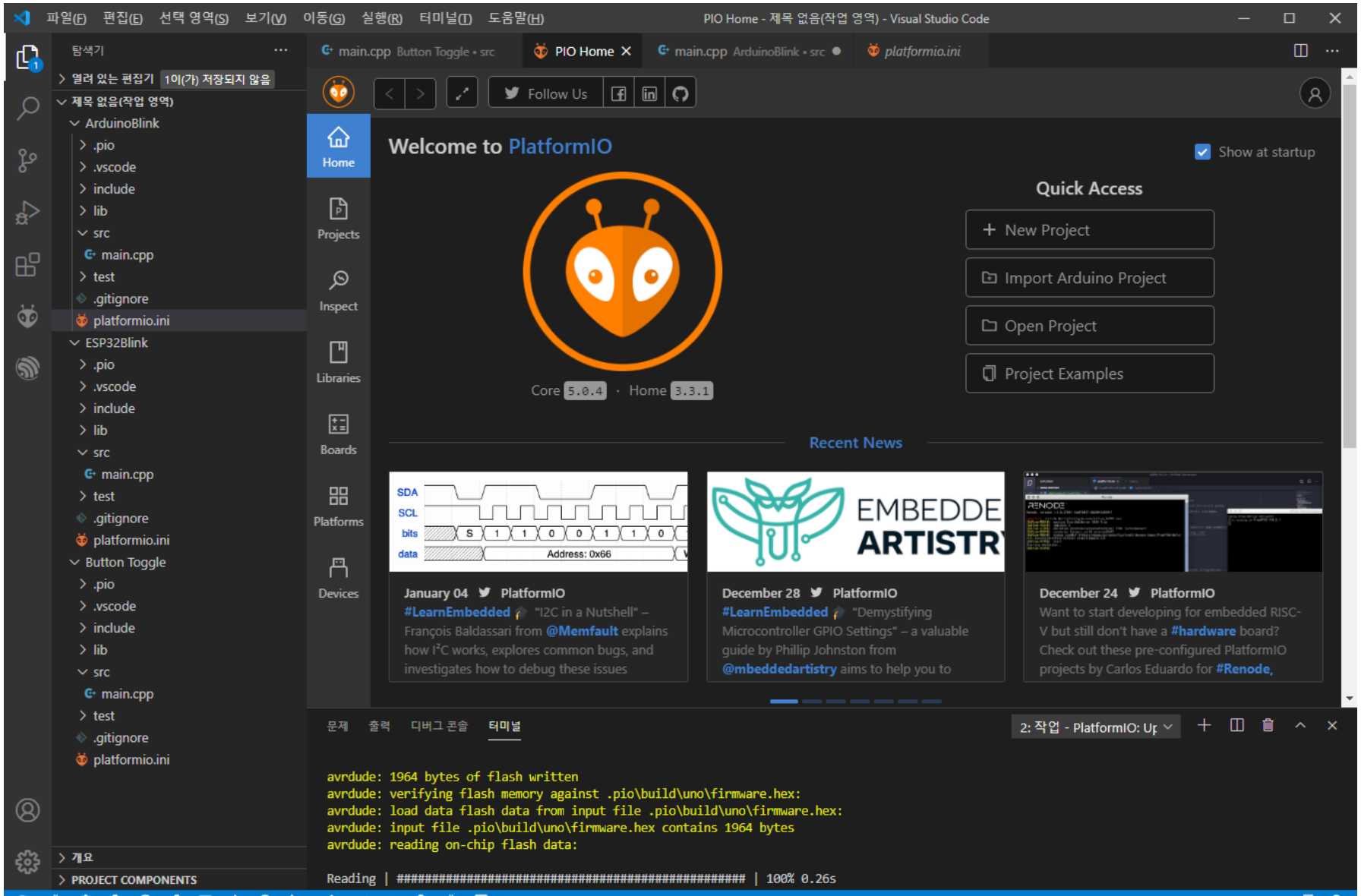


아두이노 호환보드 설정


장치 관리자
파일(F) 동작(A) 보기(V) 도움말(H)
Nebraska
DVD/CD-ROM 드라이브
IDE ATA/ATAPI 컨트롤러
기타 장치
USB2.0-Serial
네트워크 어댑터
디스크 드라이브
디스플레이 어댑터
마우스 및 기타 포인팅 장치
모니터
범용 직렬 버스 컨트롤러
AMD USB 3.10 확장 가능한 호스트 컨트롤러 - 1.10(Microsoft)
AMD USB 3.10 확장 가능한 호스트 컨트롤러 - 1.10(Microsoft)
USB Composite Device
USB Composite Device
USB 루트 허브(USB 3.0)
USB 루트 허브(USB 3.0)
USB 인쇄 지원
USB 인쇄 지원
알 수 없는 USB 장치(장치 설명자 요청 실패)
일반 USB 허브
보안 장치
사운드, 비디오 및 게임 컨트롤러
소프트웨어 구성 요소
소프트웨어 장치
시스템 장치

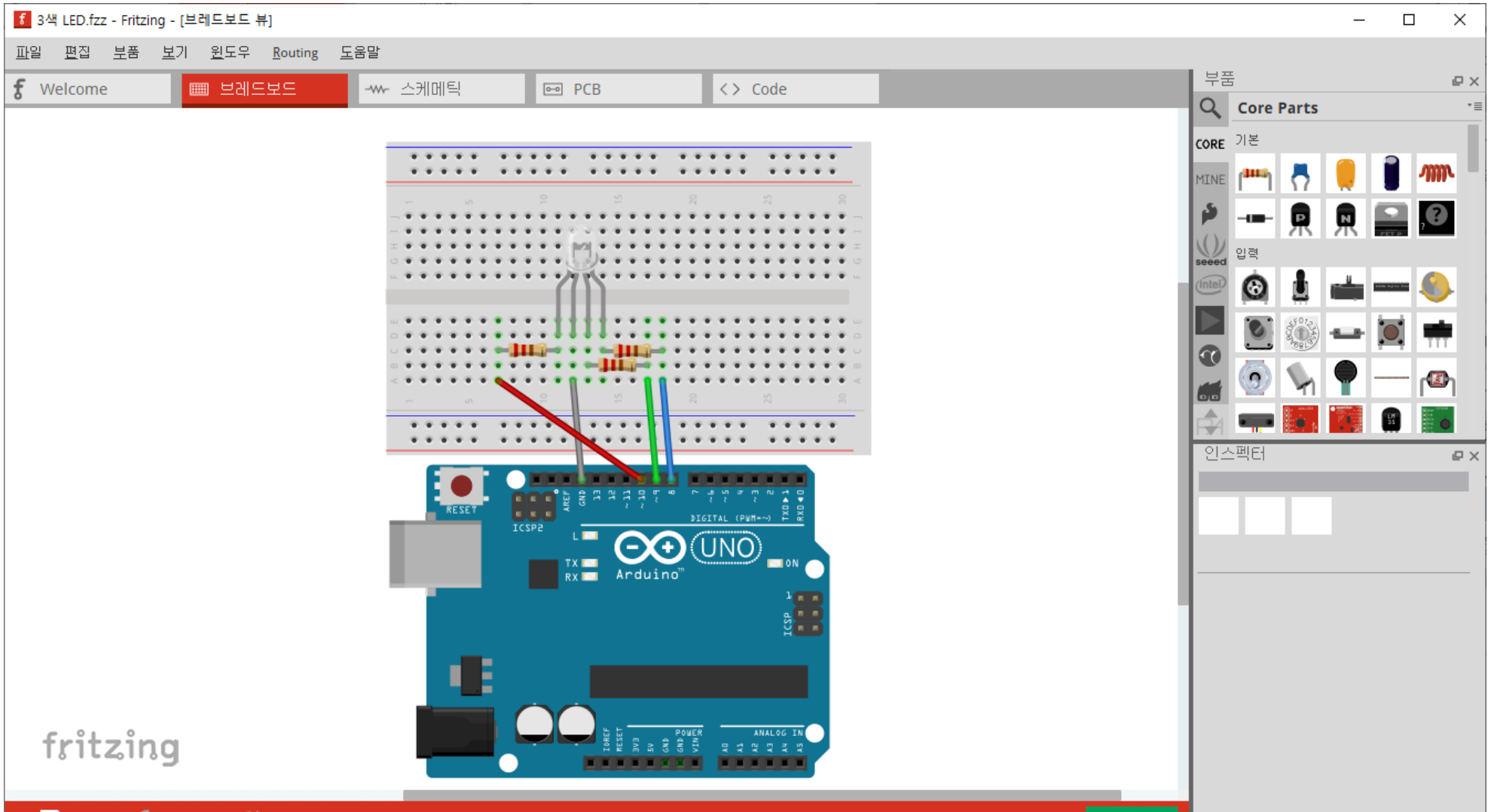

장치 관리자
파일(F) 동작(A) 보기(V) 도움말(H)
USB Composite Device
USB 루트 허브(USB 3.0)
USB 루트 허브(USB 3.0)
USB 인쇄 지원
USB 인쇄 지원
알 수 없는 USB 장치(장치 설명자 요청 실패)
일반 USB 허브
보안 장치
사운드, 비디오 및 게임 컨트롤러
소프트웨어 구성 요소
소프트웨어 장치
시스템 장치
오디오 입력 및 출력
이미징 장치
인쇄 대기열
저장소 컨트롤러
컴퓨터
키보드
펌웨어
포트(COM & LPT)
USB-SERIAL CH340(COM3)
통신 포트(COM1)
프로세서
프린터
휴먼 인터페이스 장치

개발도구 – VScode + PlatformIO

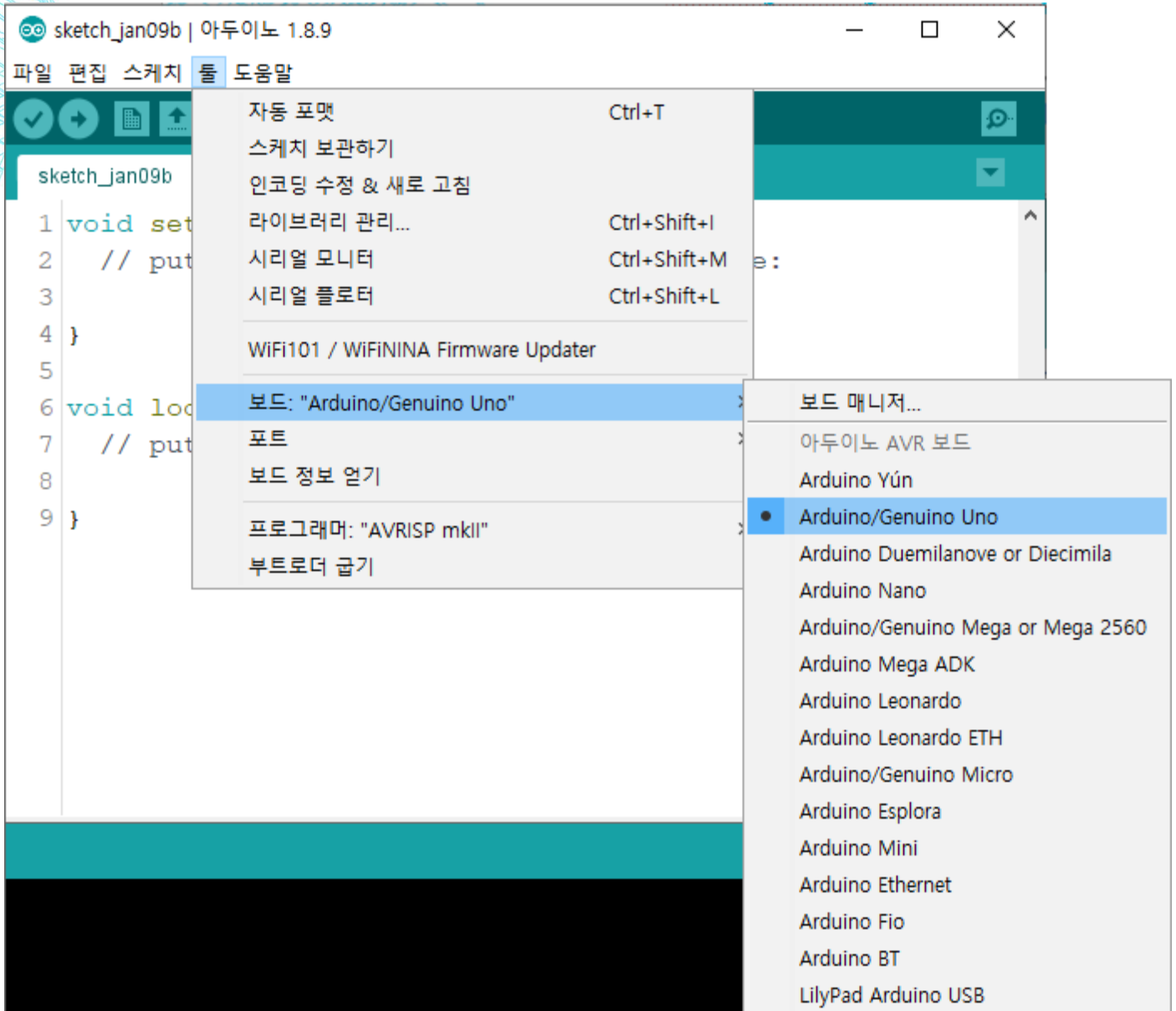


개발도구 – Fritzing

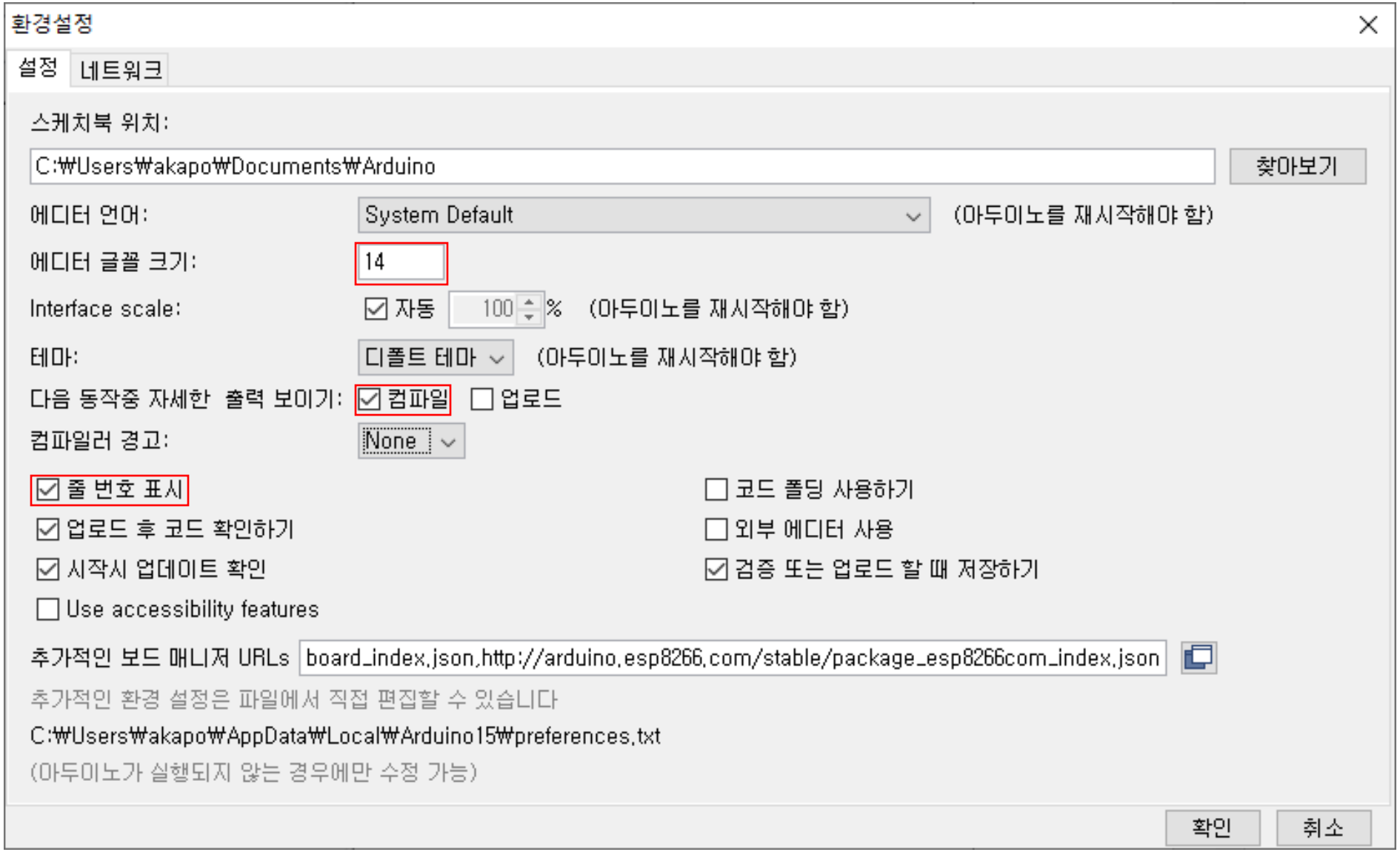
- 다운로드 : https://arduino.datahub.pe.kr/sw/fritzing.0.9.3b.64.pc.zip



개발도구 – 아두이노 스케치

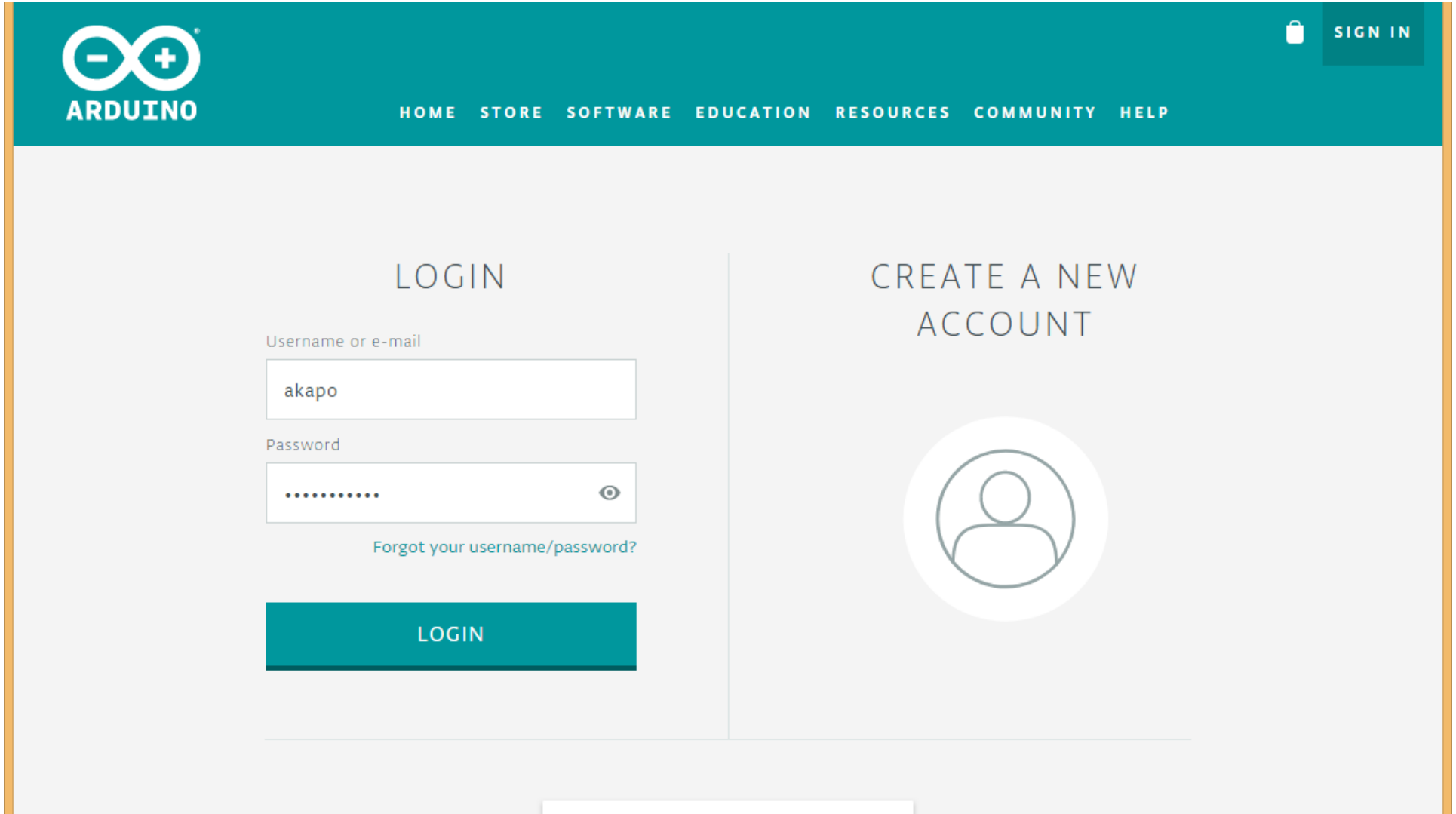


개발도구 – 아두이노 스케치



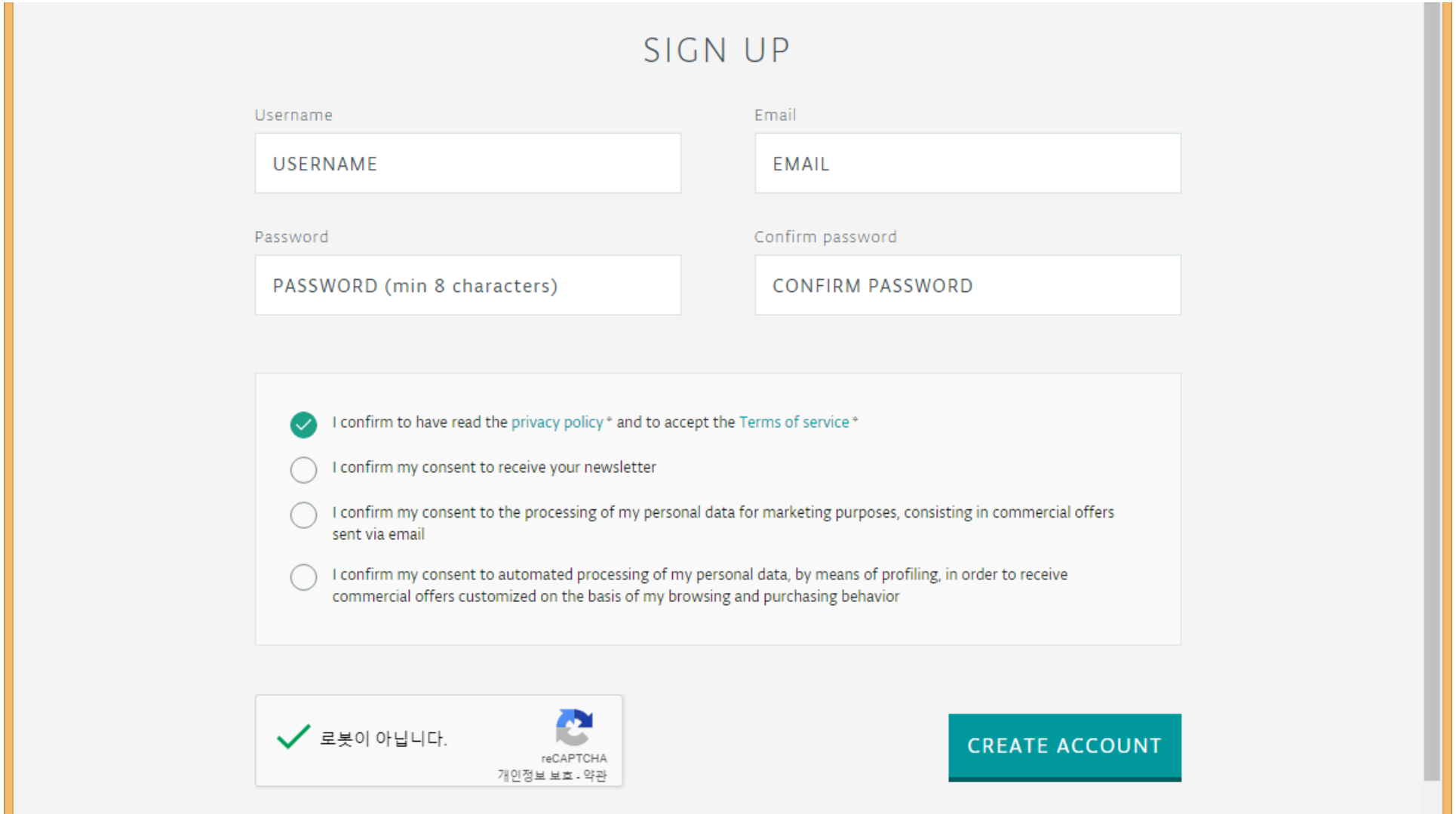
아두이노 온라인 IDE

- URL: https://create.arduino.cc/editor



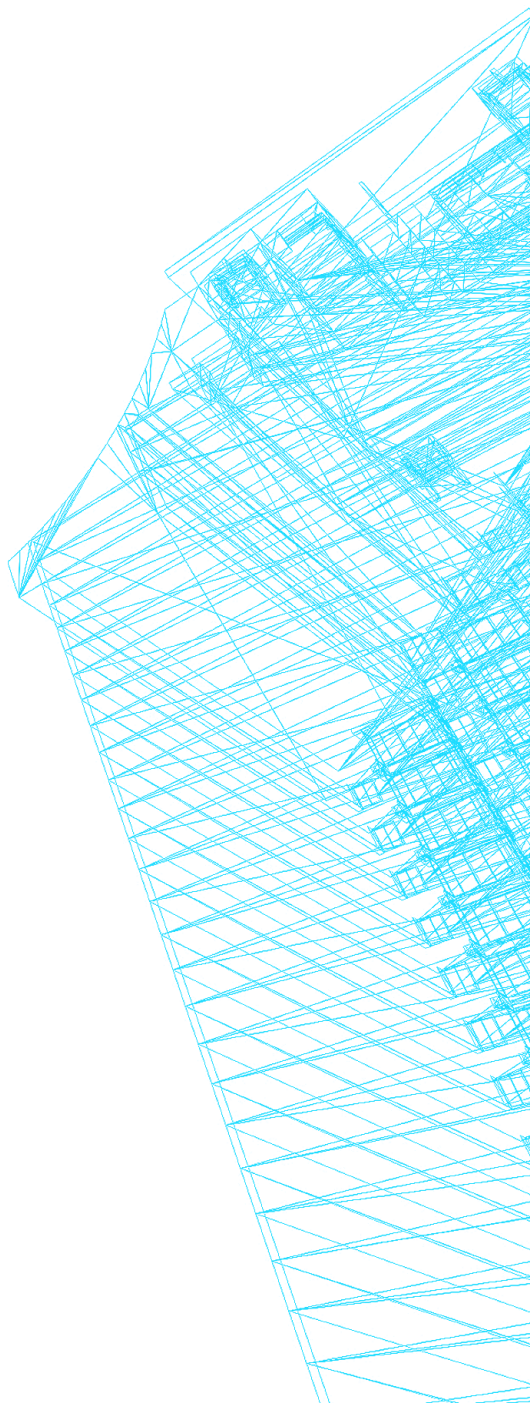
아두이노 온라인 IDE

- 회원가입



Digital Output

- LED 켜기
- LED와 저항
- Source 와 Sink
- 3색 LED



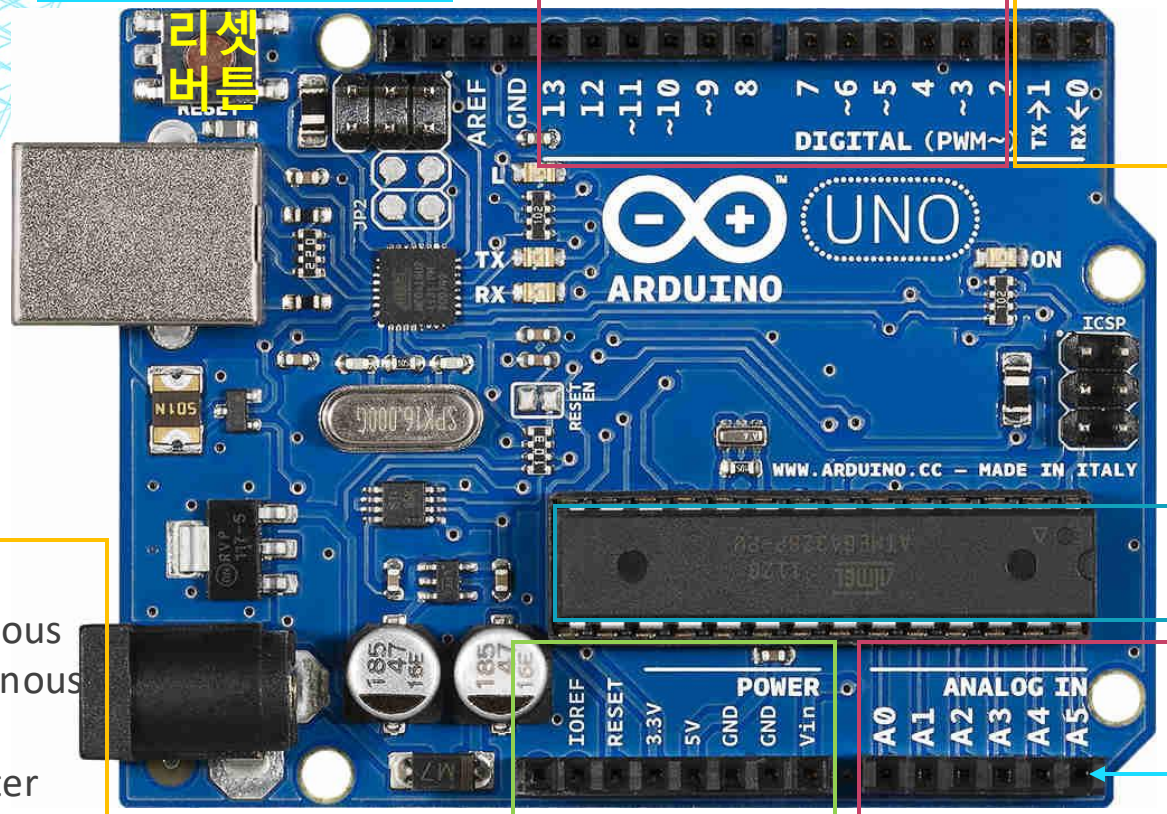
아두이노 UNO (DIP) 핀 아웃

모든 GPIO 핀
디지털 입출력 가능

디지털 출력
아날로그 출력(~)

USART
통신

- GPIO : General Purpose Input Output
- 입력 : 외부에서 아두이노 안으로 들어오는 신호
- 출력 : 아두이노 내부에서 밖으로 나가는 신호
  - 디지털 출력 : 출력 값이 0(LOW, 0V), 1(HIGH, 5V) 이렇게 두 가지 상태만 있는 출력.
  - 아날로그 출력 : 출력 값을 0% ~ 100% (0V ~ 5V)까지 단계별로 조절할 수 있는 출력.



Universal
Synchronous
Asynchronous
Receiver
Transmitter

ATMEGA 328P
MCU 칩

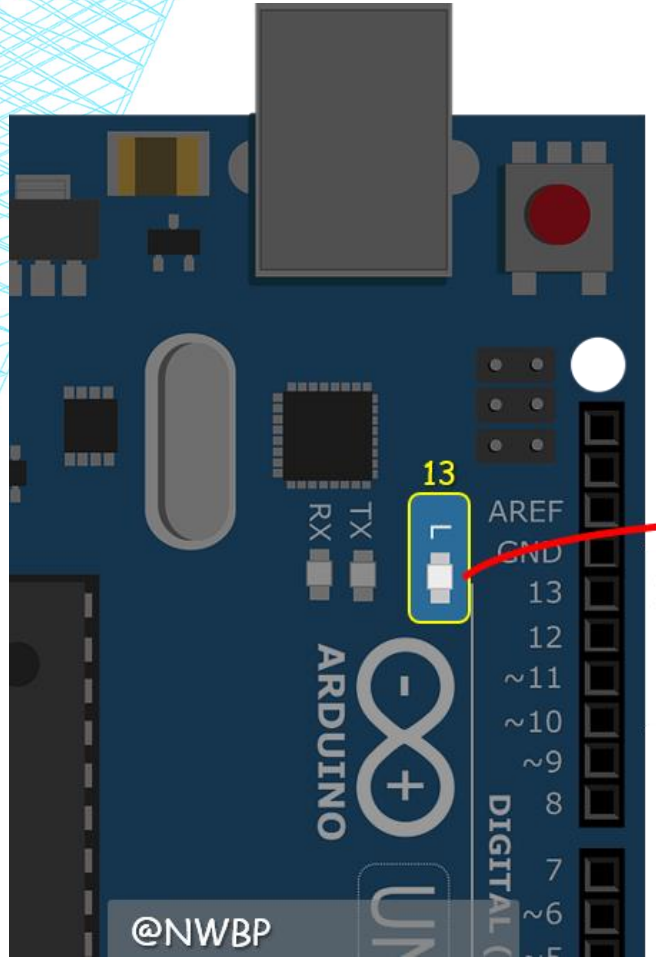
전원

아날로그 입력

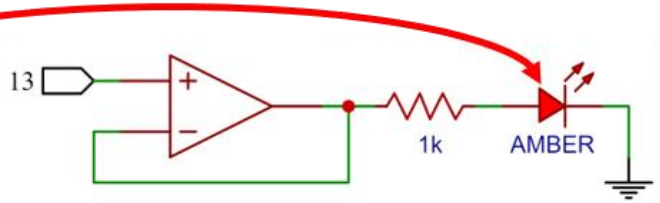
핀
or
포트

내장 LED (Built in LED)

- 회로



HIGH: 5V
LOW: 0V



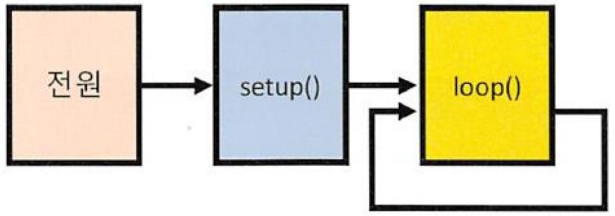
13 HIGH --> LED ON

13 LOW --> LED OFF

- 프로그램

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(500);

  digitalWrite(LED_BUILTIN, LOW);
  delay(500);
}
```

내장 LED (Built in LED)



- 프로그램 분석

1-1.ino

```
void setup() {
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);
  //digitalWrite(13, 1);
  //digitalWrite(13, true);
  delay(500);

  digitalWrite(13, LOW);
  //digitalWrite(13, 0);
  //digitalWrite(13, false);
  delay(500);
}
```

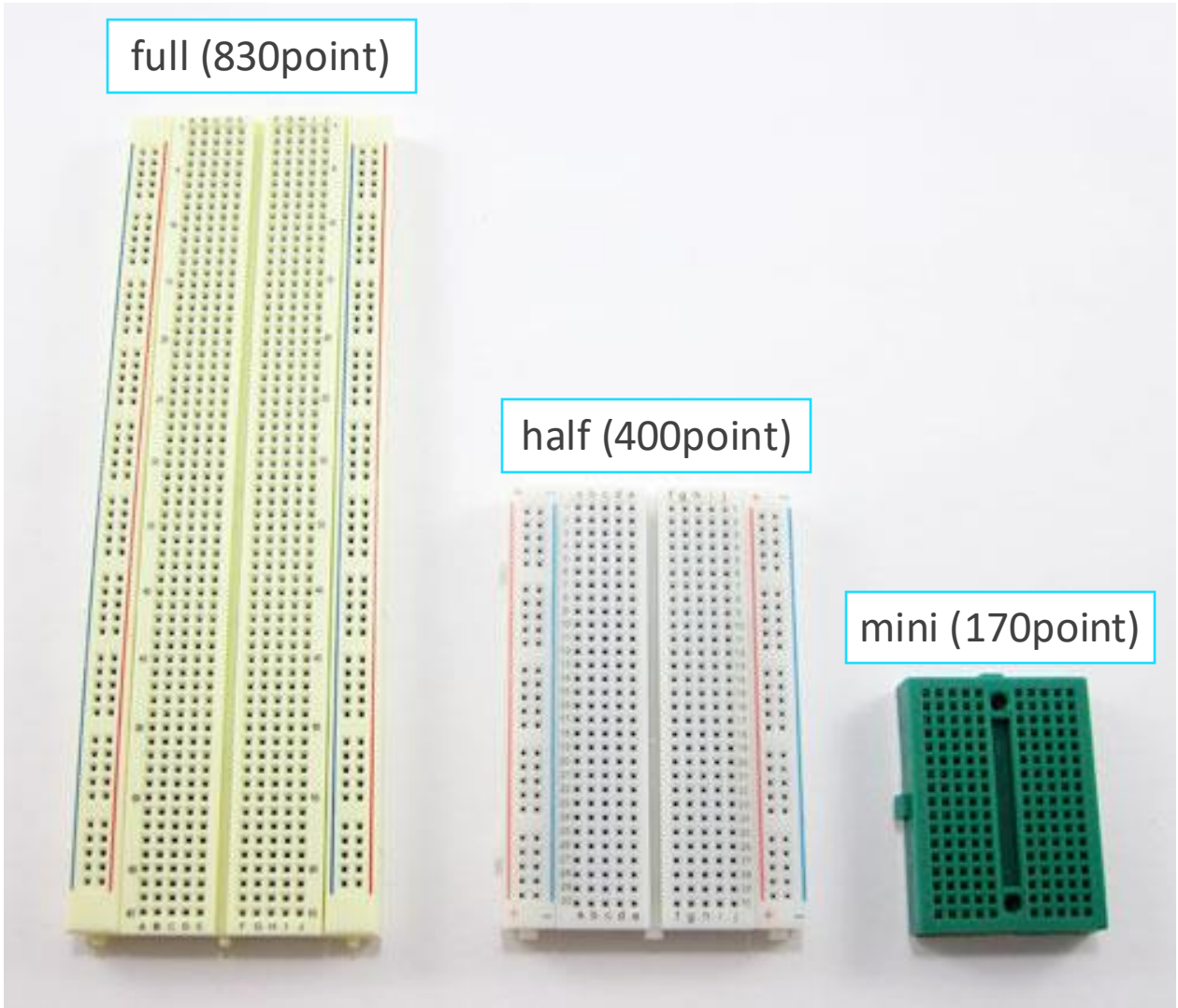
핀 모드를 설정
pinMode(핀번호, 모드)

디지털 출력
digitalWrite(핀번호, 상태)

지연
dealy(밀리초)

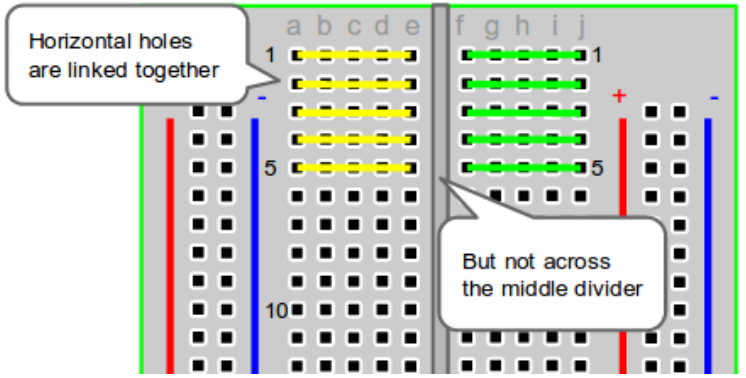
브레드보드(Breadboard – 빵판)

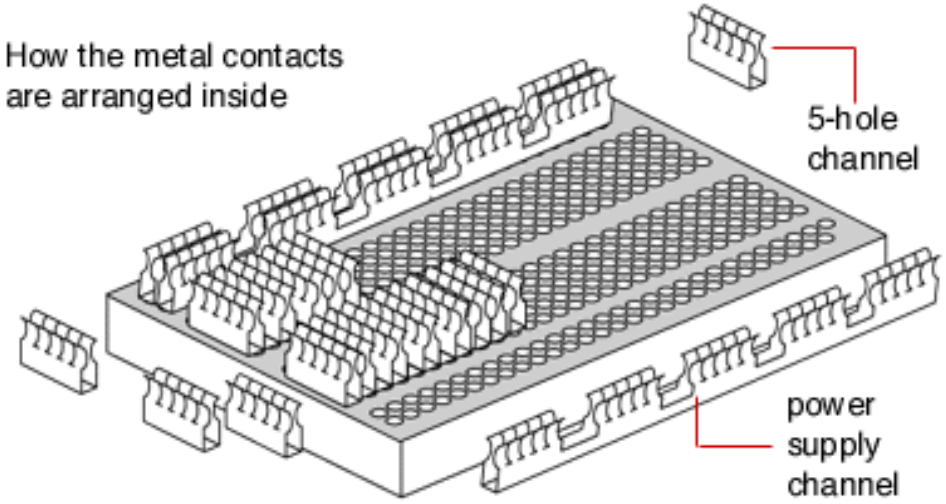
- 전기 인두를 이용해 납땜을 하지 않아도 전자회로 구성 가능

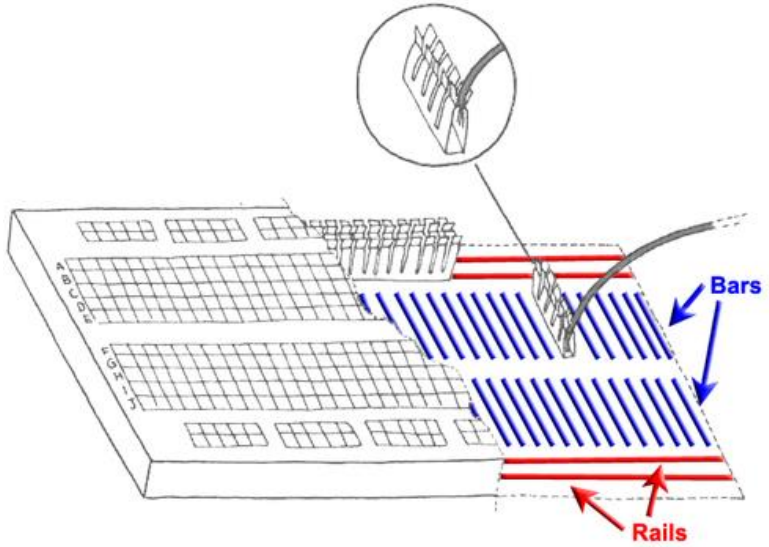


브레드보드(Breadboard – 빵판)

- 내부구조

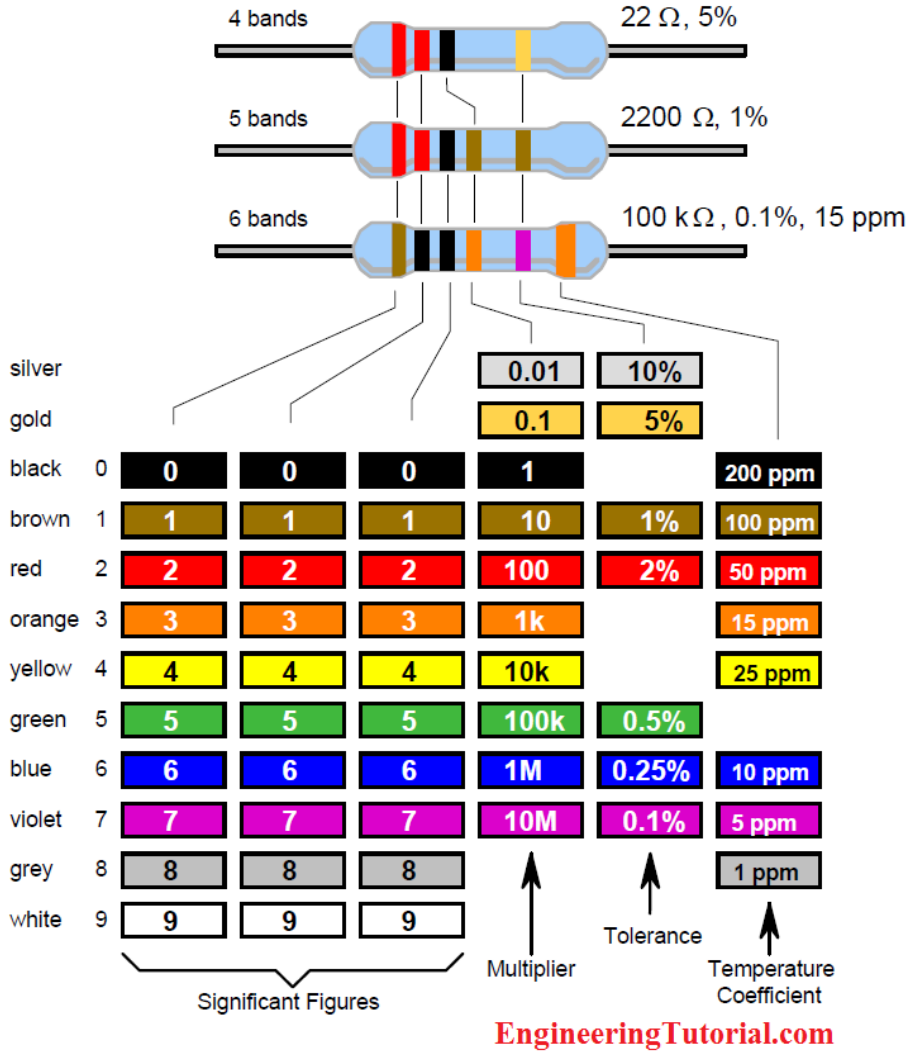






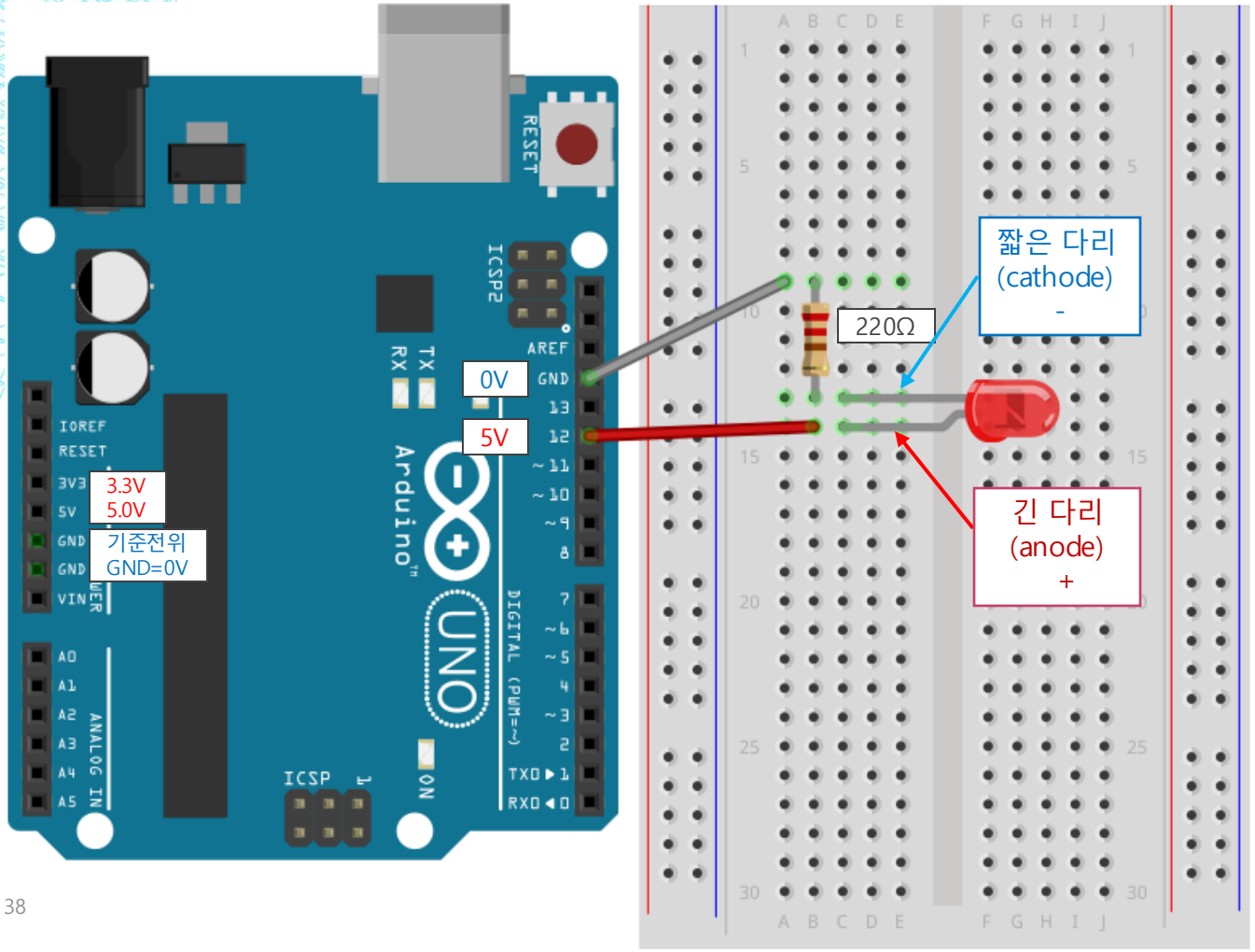
저항

- 사용 이유
  - 회로에 흐르는 전류량을 제한하기 위해
- 컬러코드
  - 저항에 색상 띠로 저항 수치를 표시
- 저항 값 계산기
  - http://kor.pe.kr/util/4/r_calc/



LED 온/오프

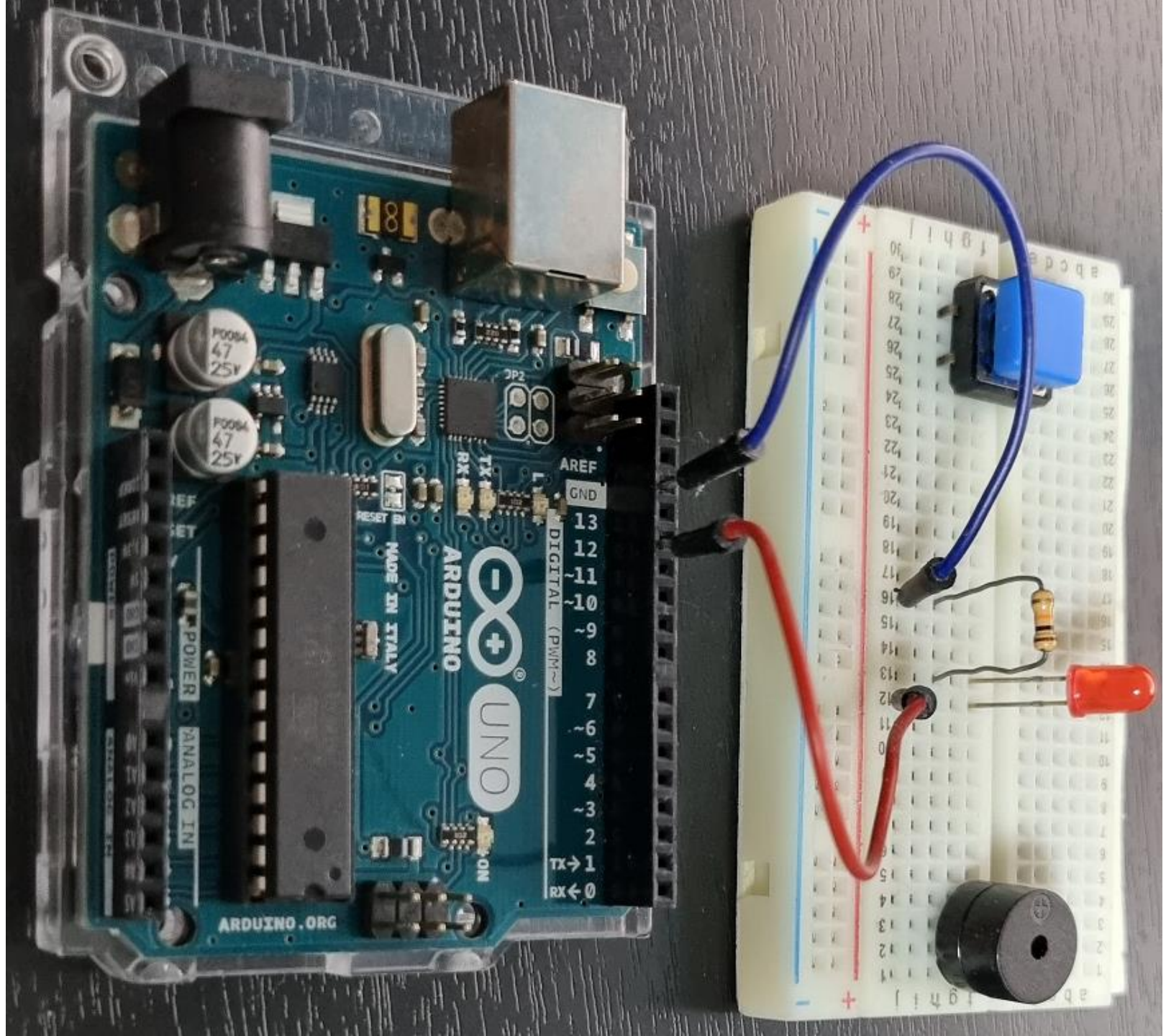
- 회로



- 프로그램

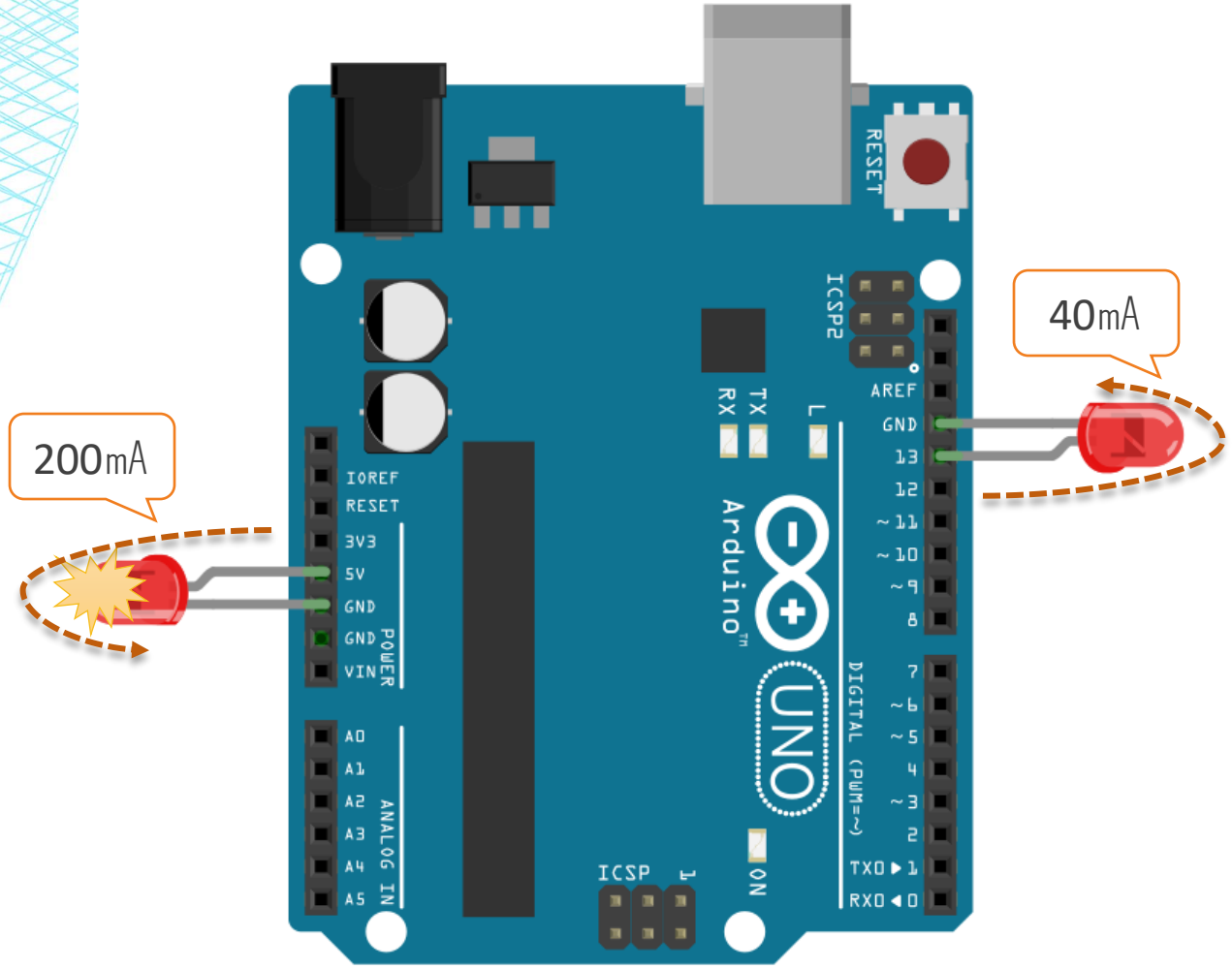
1-2.ino

```
#define LED 12

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(LED, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  digitalWrite(LED, HIGH);
  delay(500);
  digitalWrite(LED_BUILTIN, LOW);
  digitalWrite(LED, LOW);
  delay(500);
}
```


ARDUINO
UNO
ARDUINO.ORG
MADE IN ITALY
POWER
ANALOG IN
DIGITAL (PWM~)
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
RESET
ON

최대 전류

- 회로 구성



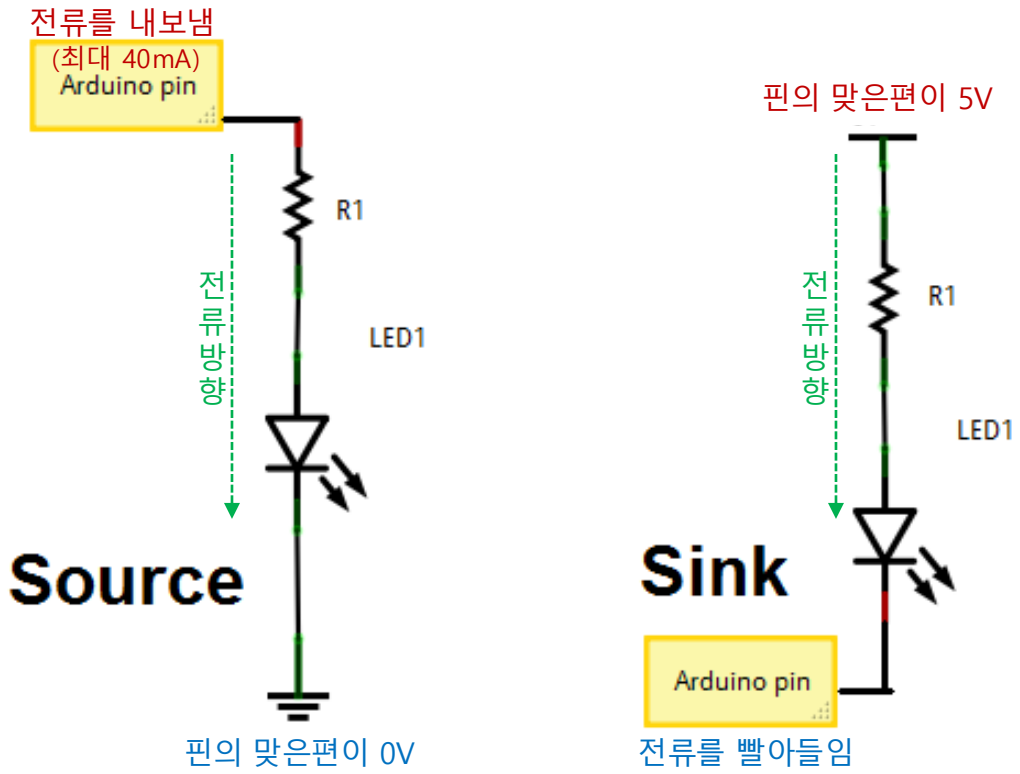
Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground	-0.5V to V_{CC}+0.5V
Voltage on $\overline{\text{RESET}}$ with respect to Ground	-0.5V to +13.0V
Maximum Operating Voltage	6.0V
DC Current per I/O Pin	40.0 mA
DC Current V_{CC} and GND Pins	200.0 mA

전류의 방향

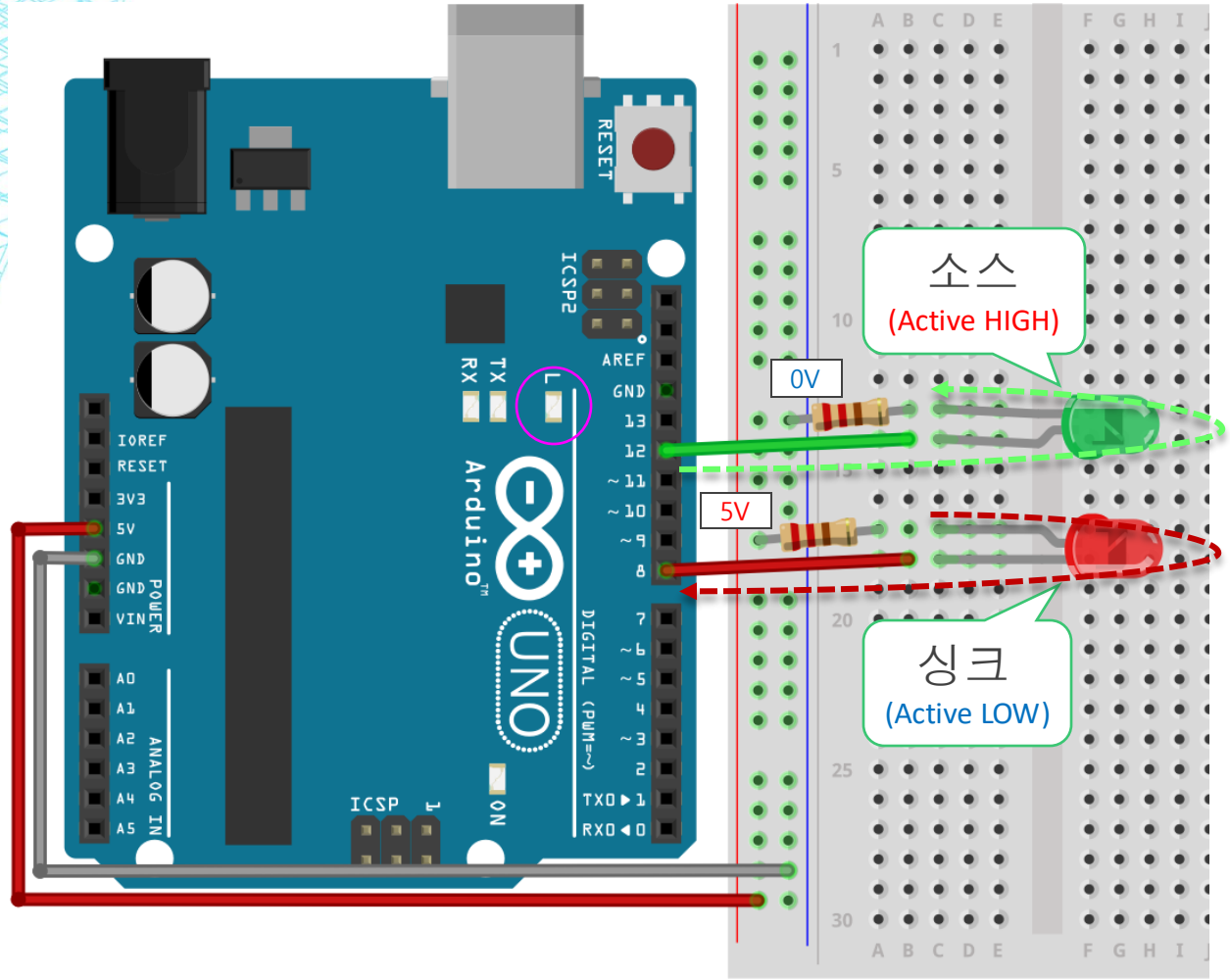
항상 HIGH 가 ON이고 LOW가 OFF인 것은 아니다.
전류의 방향을 조절하여 LOW 일 때 ON 이 되도록 만들 수도 있다.

- 소스
  - 칩에서 나가는 전류를 사용
  - HIGH 일 때 ON이 됨
  - Active HIGH
- 싱크
  - 칩으로 들어오는 전류를 사용
  - LOW일 때 ON 이 됨
  - Active LOW
  - 큰 전류를 필요로 할 때 사용 (IO포트의 전류 제한에 걸리지 않으므로...)



소스 vs 싱크

- 회로 연결



1-3.ino

```
#define LED_R 8
#define LED_G 12

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(LED_R, OUTPUT);
  pinMode(LED_G, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  digitalWrite(LED_R, HIGH);
  digitalWrite(LED_G, HIGH);
  delay(1000);

  digitalWrite(LED_BUILTIN, LOW);
  digitalWrite(LED_R, LOW);
  digitalWrite(LED_G, LOW);
  delay(1000);
}
```

내장 LED를 HIGH인지 LOW인지 알려주는 인디케이터로 보면됨

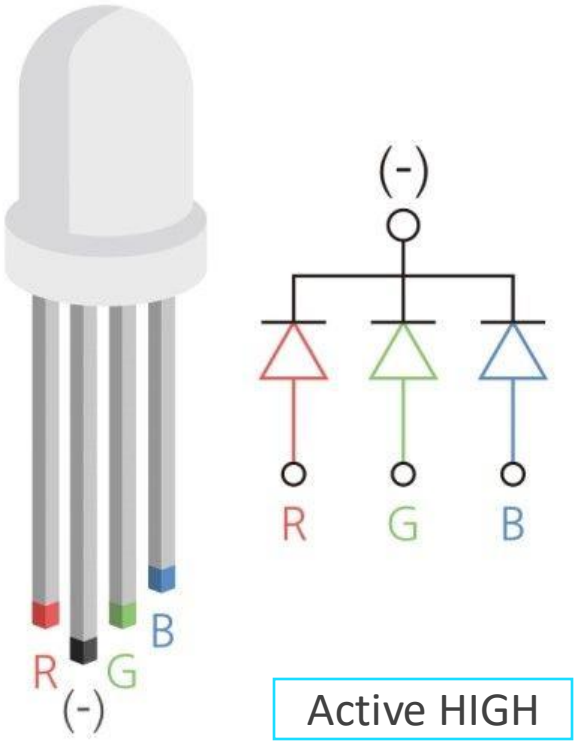
3색 LED

- 모양

엥? 왜 다리가 네개지?

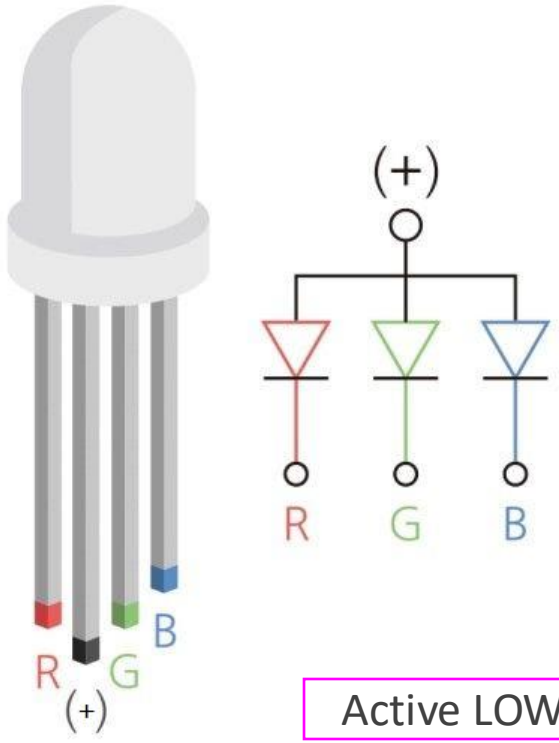
3색이면 다리가 6개 아닌가?

Common Cathode (-) 공통 음극



Active HIGH

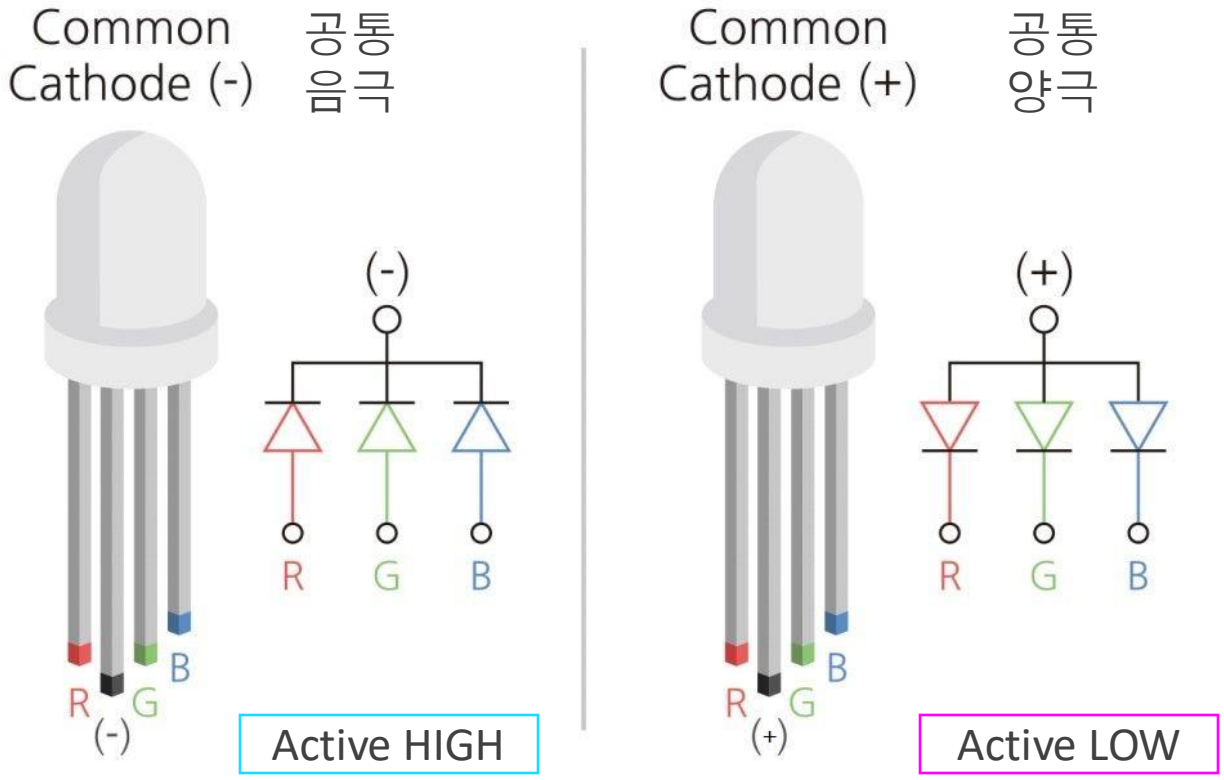
Common Cathode (+) 공통 양극



Active LOW

3색 LED 퀴즈1

- 내가 가진 3색 LED는 공통 음극인가? 공통 양극인가?
- 어떻게 알아낼 수 있을까?



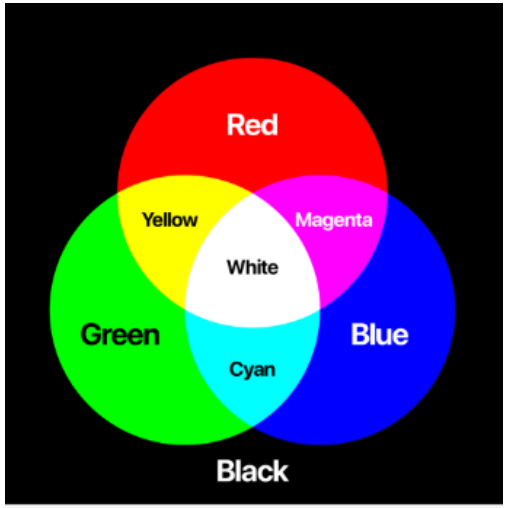
3색 LED 퀴즈2

- 공통 음극(Common cathode) 3색 LED를 10번 핀에 빨강, 9번 핀에 녹색, 8번 핀에 파랑을 연결하고,

1) 빨간불, 녹색불, 파란불을 1초 간격으로 출력하시오.

2) 빛의 3원색 조합을 이용하여 8가지 색상을 각 1초씩 출력하는 프로그램을 제작하시오.

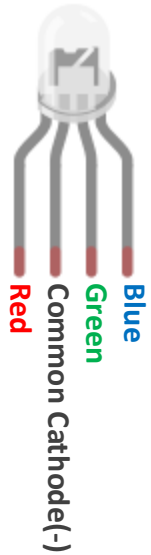
- 색상표



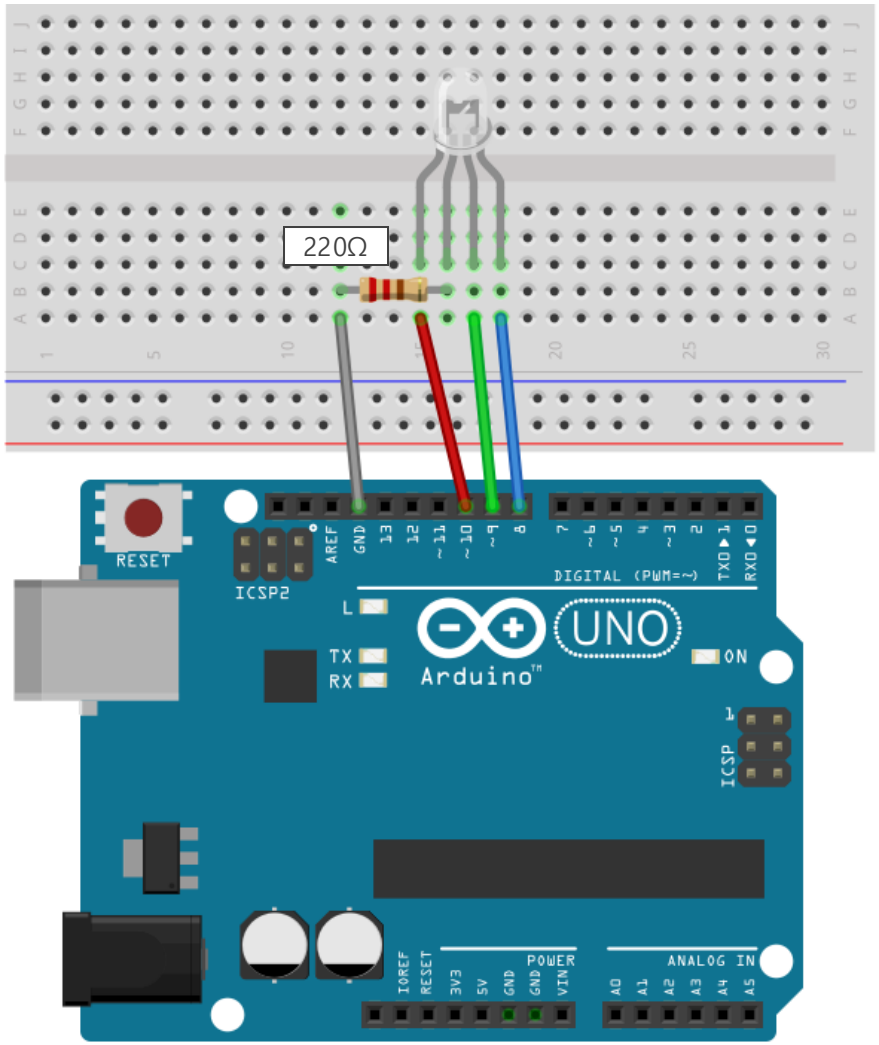
RGB 벤 다이어그램

	R	G	B
빨강 (Red)	HIGH	LOW	LOW
녹색 (Green)	LOW	HIGH	LOW
파랑 (Blue)	LOW	LOW	HIGH
노랑 (Yellow)	HIGH	HIGH	LOW
자주 (Magenta)	HIGH	LOW	HIGH
청록 (Cyan)	LOW	HIGH	HIGH
흰색 (White)	HIGH	HIGH	HIGH

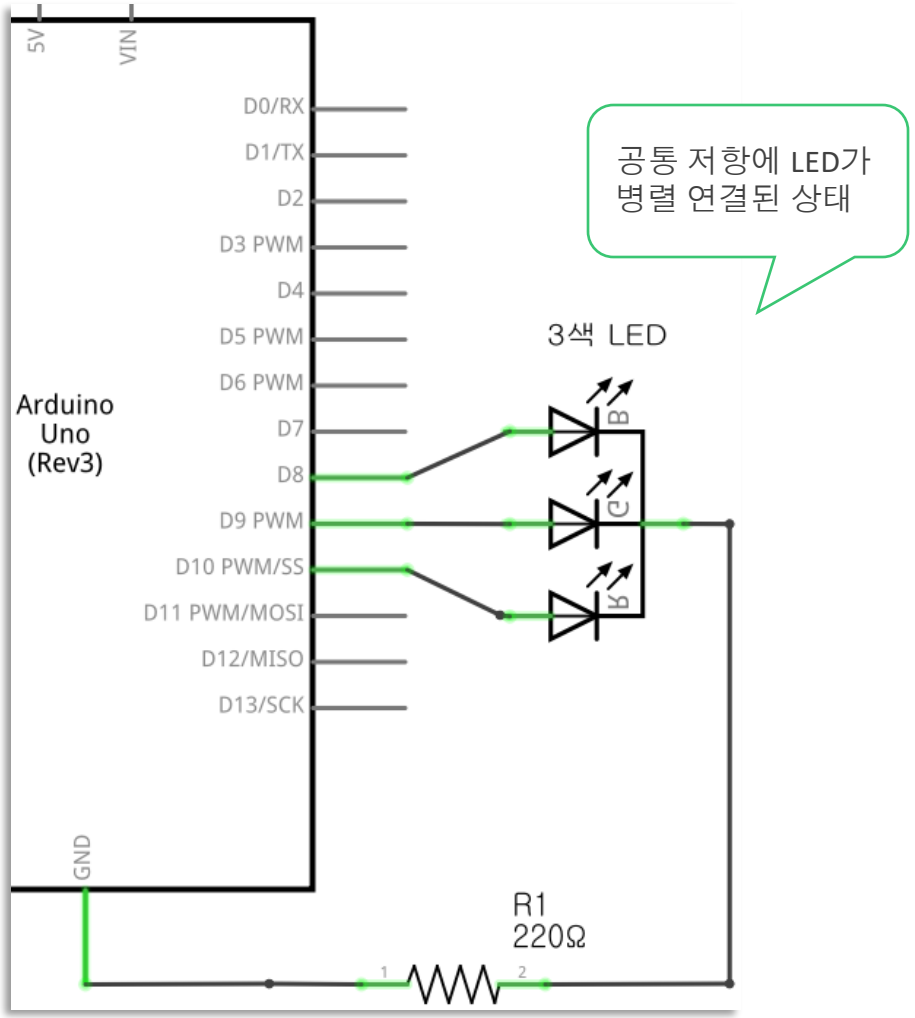
3색 LED Quiz



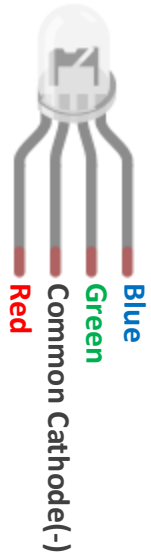
- 회로 구성 방법 1



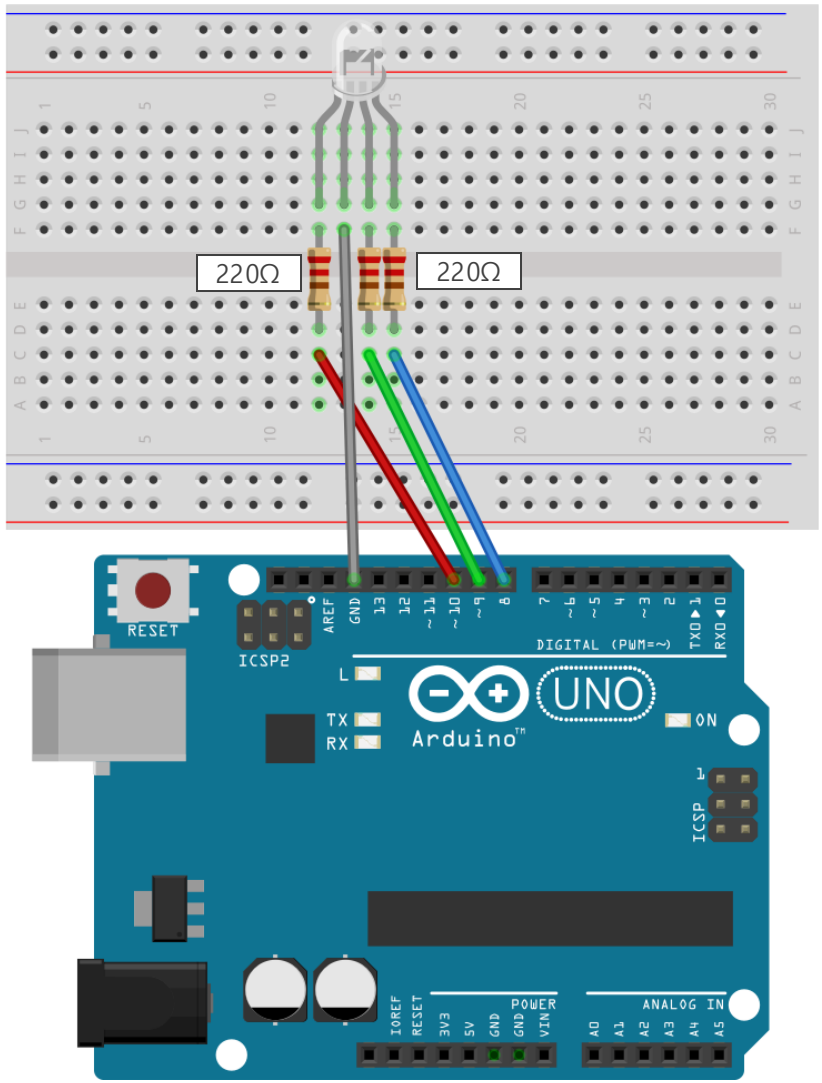
- 회로도



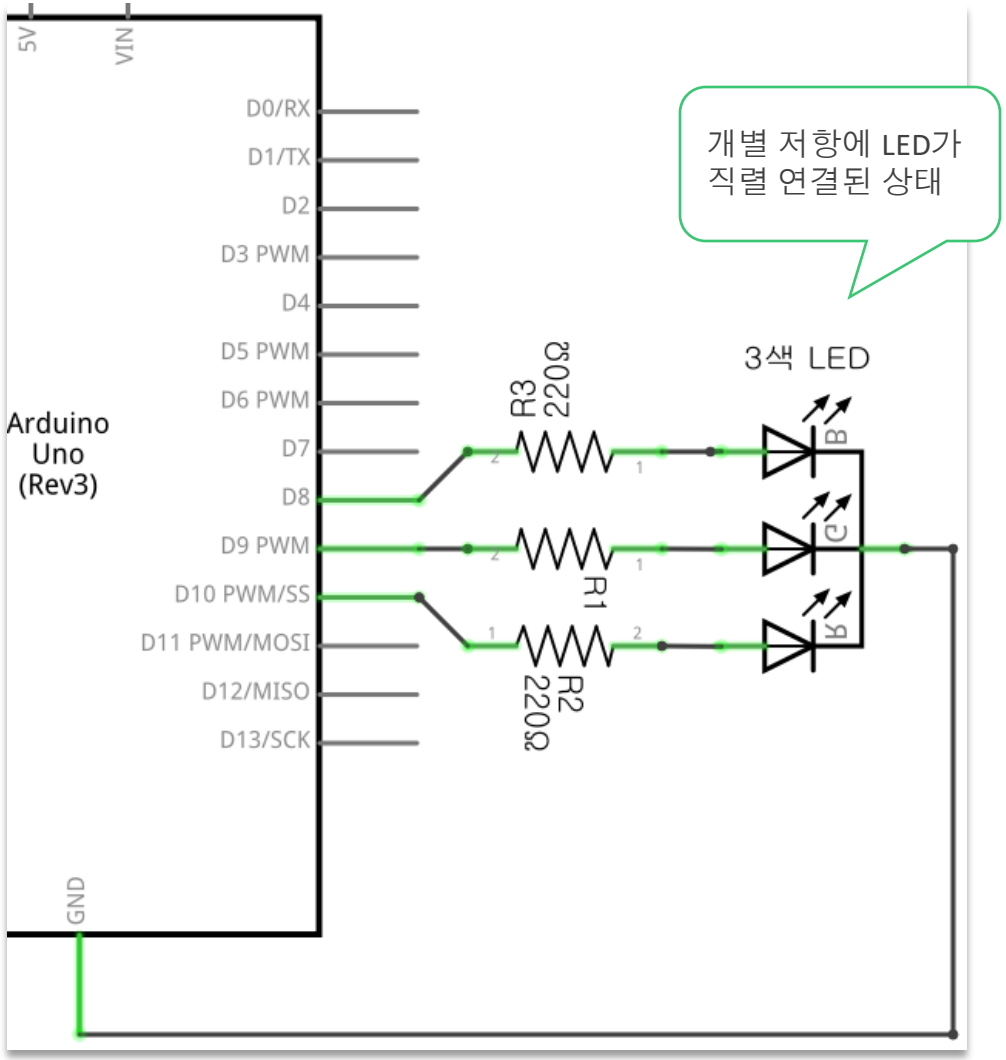
3색 LED Quiz



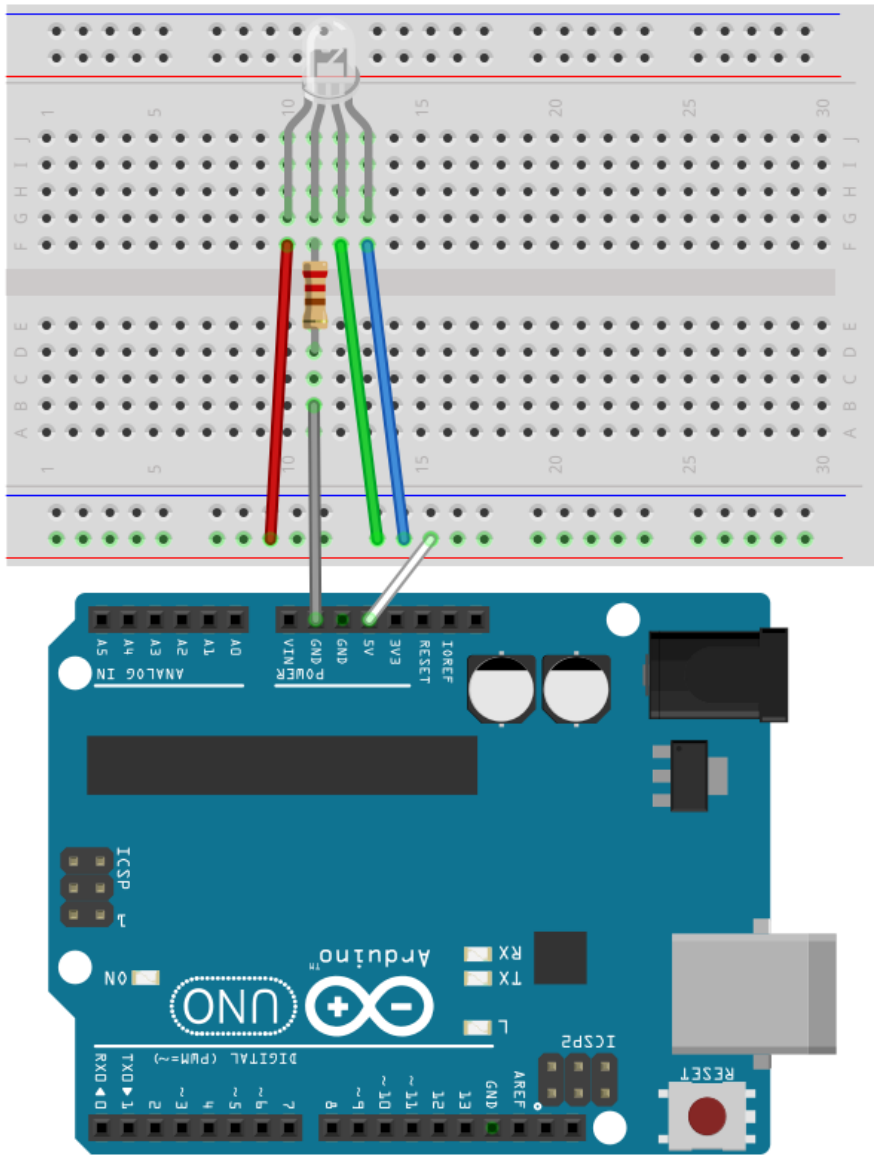
- 회로 구성 방법 2



- 회로도

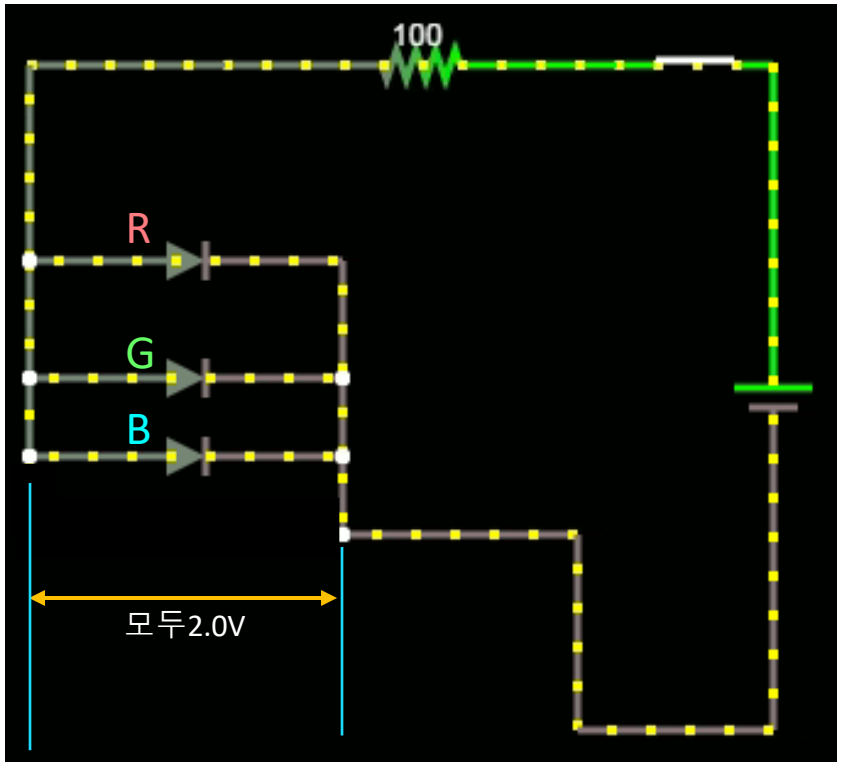


공통 저항에 LED 병렬 연결

- 

1:1 연결 시
LED의 V_f값
R: 2.0V
G: 2.7V
B: 2.8V
(R220에서
실측정치임)

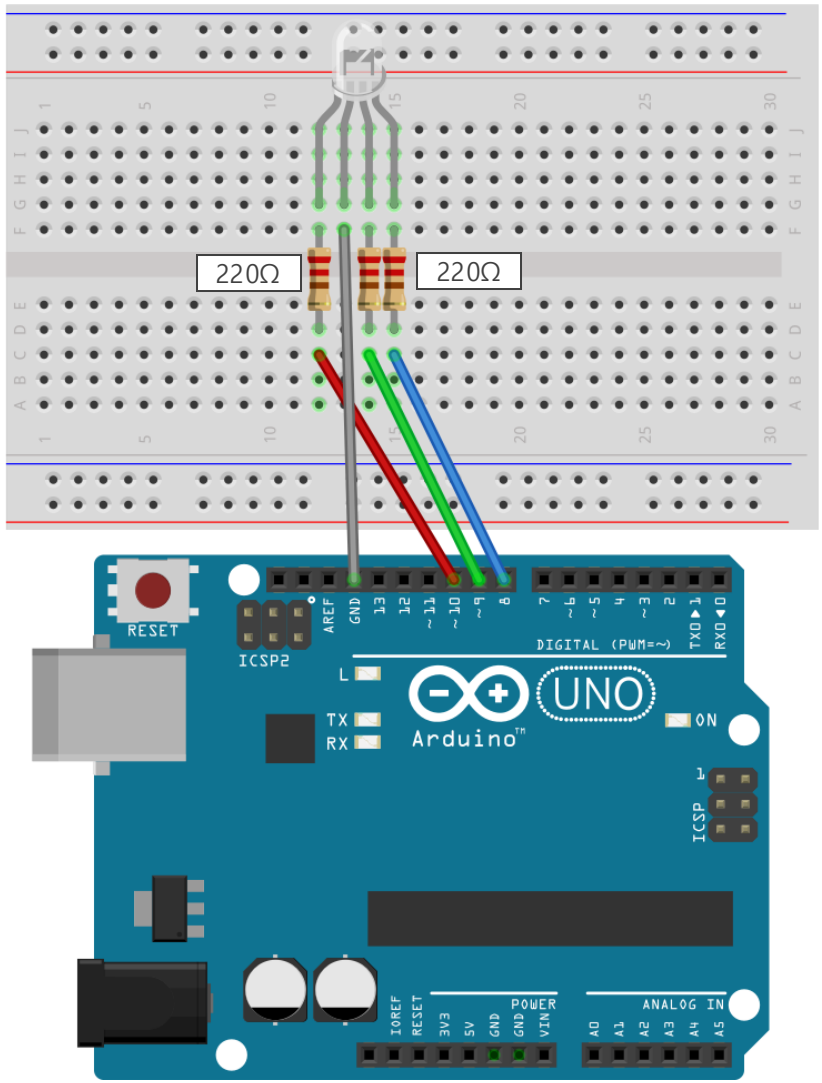
- 다른 특성의 LED 사용



- 모든 LED에 동일한 전압이 걸린다.
- 통과하기 쉬운(V_f가 낮은) LED로 대부분의 전류가 흐른다.

3색 LED Quiz 정답

- 회로 구성

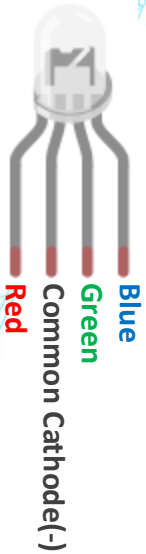


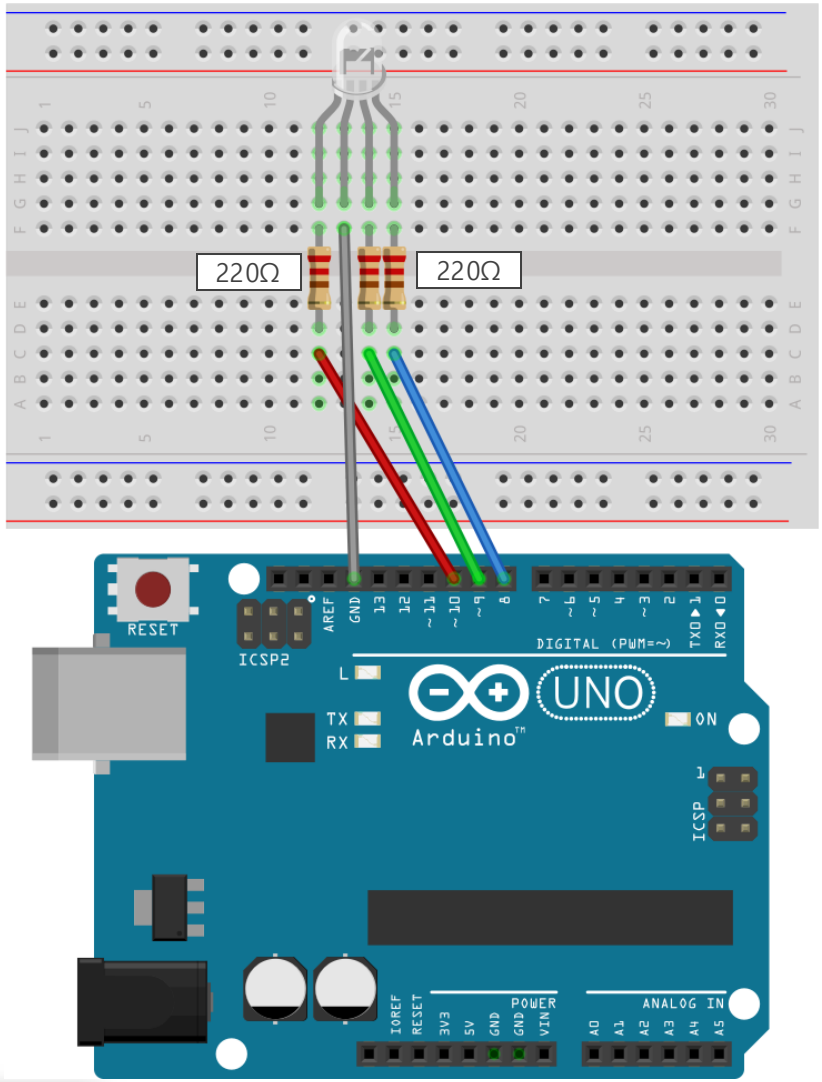
1-4.ino

```
// 3색 켜기
#define RED   10
#define GREEN 9
#define BLUE  8

void setup() {
  pinMode(RED, OUTPUT);
  pinMode(GREEN, OUTPUT);
  pinMode(BLUE, OUTPUT);
}

void loop() {
  digitalWrite(RED, HIGH);     // 빨간불 켜기
  digitalWrite(GREEN, LOW);
  digitalWrite(BLUE, LOW);
  delay(1000);

  digitalWrite(RED, LOW);
  digitalWrite(GREEN, HIGH);   // 녹색불 켜기
  digitalWrite(BLUE, LOW);
  delay(1000);

  digitalWrite(RED, LOW);
  digitalWrite(GREEN, LOW);
  digitalWrite(BLUE, HIGH);    // 파란불 켜기
  delay(1000);
}
```

3색 LED Quiz 정답

- 회로 구성





1-4.ino

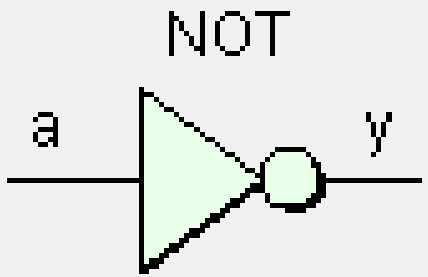
```
// 8색 켜기
#define RED   10
#define GREEN 9
#define BLUE  8

void setup() {
  for(int p=8; p<=10; p++) {
    pinMode(p, OUTPUT);
  }
}

void loop() {
  for(int c=0; c<8; c++) {
    bool r = (bool)(0b001 & c);
    bool g = (bool)(0b010 & c);
    bool b = (bool)(0b100 & c);

    digitalWrite(RED, r);
    digitalWrite(GREEN, g);
    digitalWrite(BLUE, b);
    delay(1000);
  }
}
```

No	BGR	Color
0	000	Black
1	001	Red
2	010	Green
3	011	Yellow
4	100	Blue
5	101	Magenta
6	110	Cyan
7	111	White

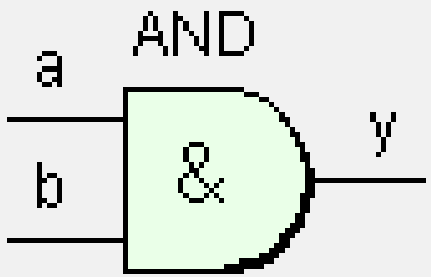
```
(R) = 0b001
(G) = 0b010
(B) = 0b100
---------
ex)
    101
  & 001 (R)

    101
  & 010 (G)

    101
  & 100 (B)
```

논리연산

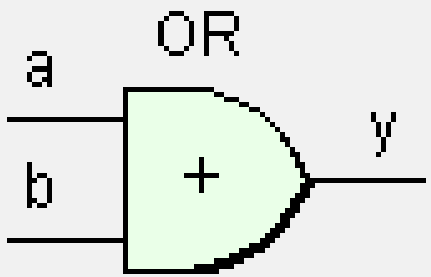
!	&	|	^



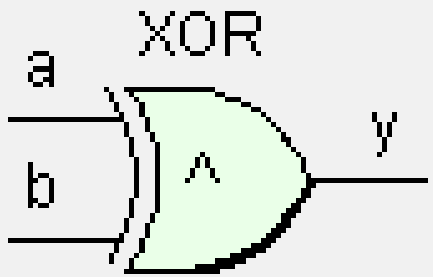
a	y
0	1
1	0



a	b	y
0	0	0
0	1	0
1	0	0
1	1	1



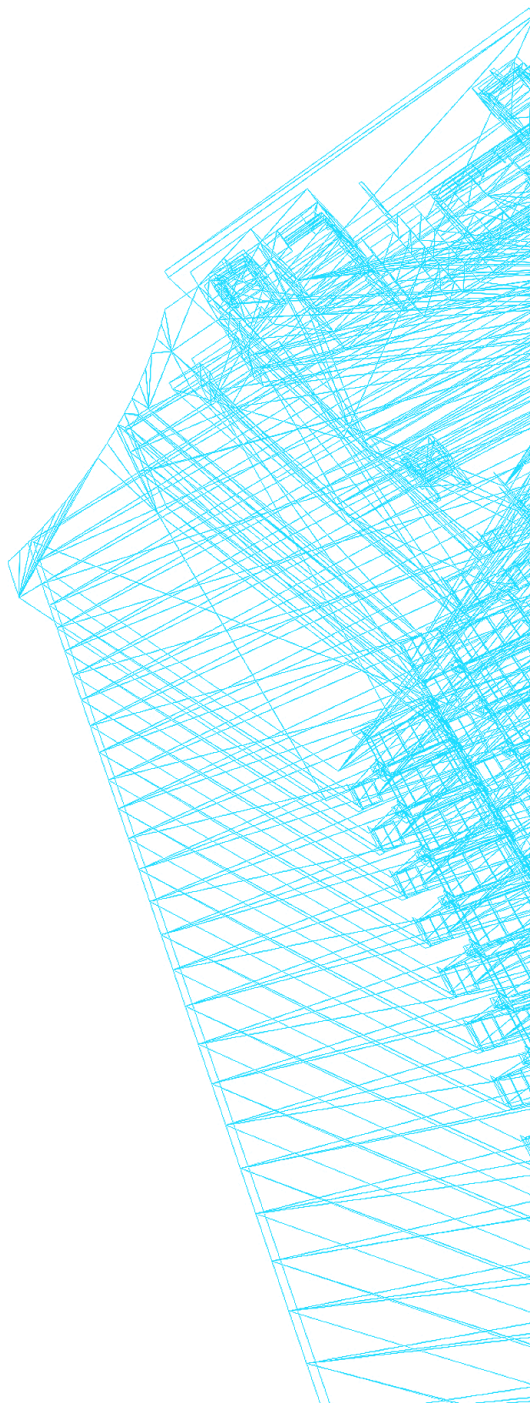
a	b	y
0	0	0
0	1	1
1	0	1
1	1	1



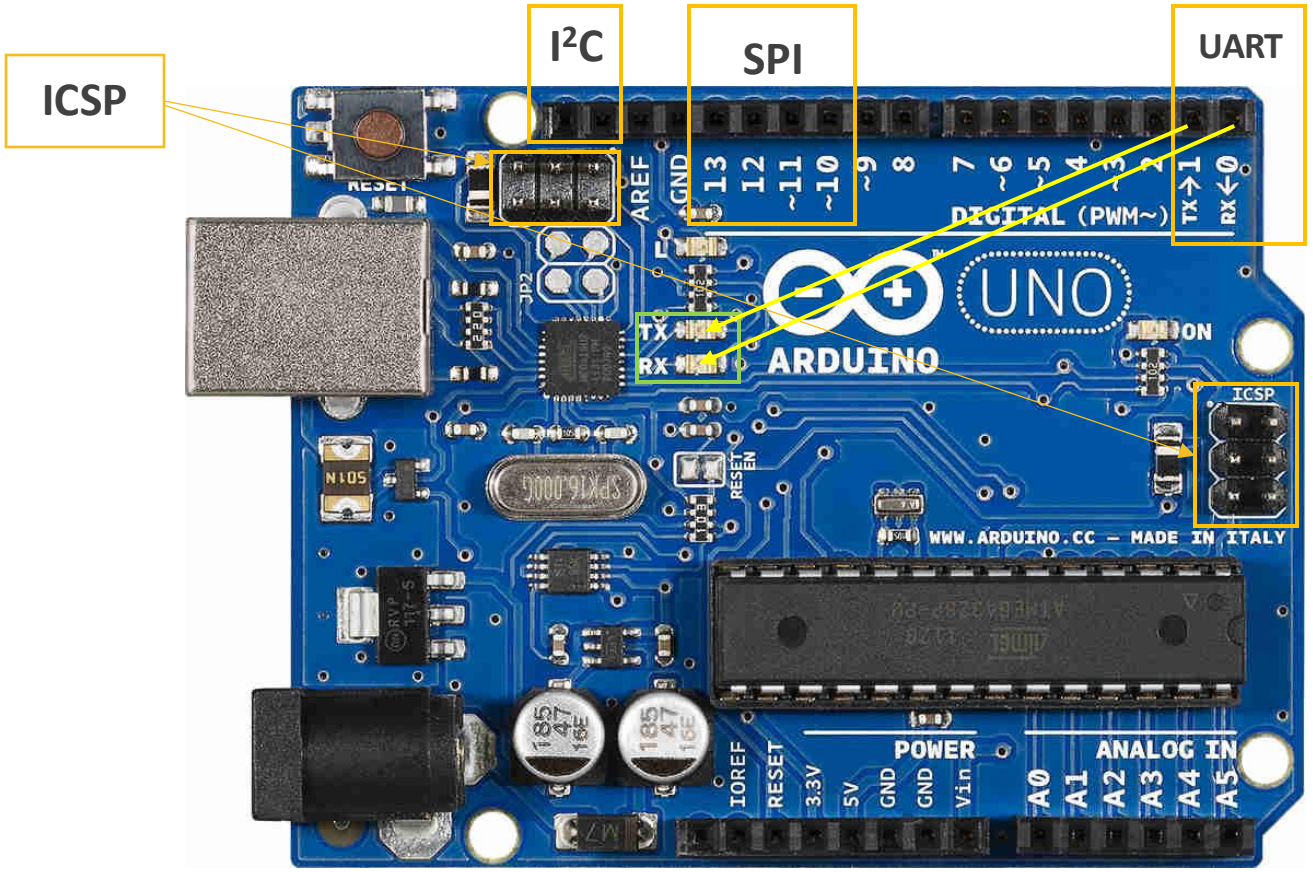
a	b	y
0	0	0
0	1	1
1	0	1
1	1	0

아두이노 시리얼 통신

- 아두이노 통신포트
- Serial 포트를 출력창으로 사용하기



아두이노 UNO 통신 포트



- UART : Universal Asynchronous Receiver and Transmitter
- I2C : Inter-Integrated Circuit
- SPI : Serial Peripheral Interface
- ICSP : In-Circuit Serial Programming

Serial 포트를 출력창으로 사용하기

n

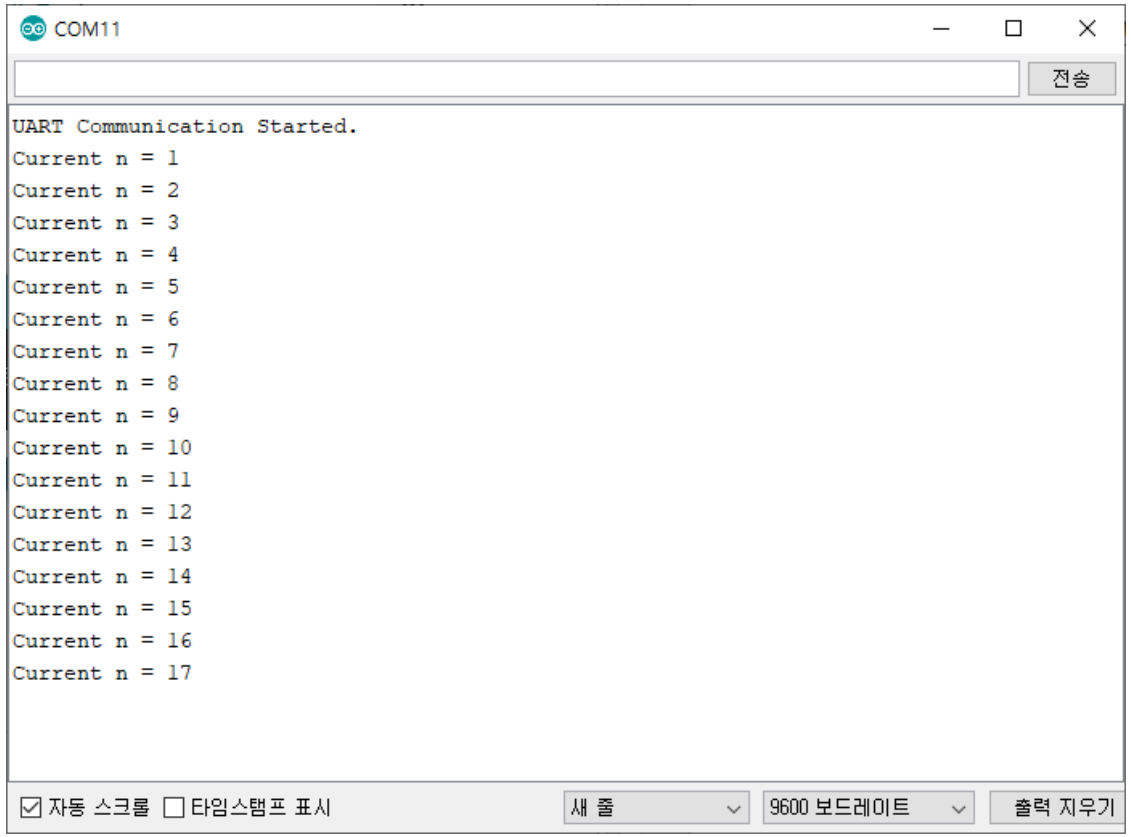
1

- 소스코드1

1-5.ino

```
int n = 1;   // 전역변수

void setup() {
  // 통신속도 9600baud로 초기화
  Serial.begin(9600);
  Serial.println("Communication Started.");
}

void loop() {
  Serial.print("Current n = ");
  Serial.println(n);
  n++;
  delay(500);
}
```

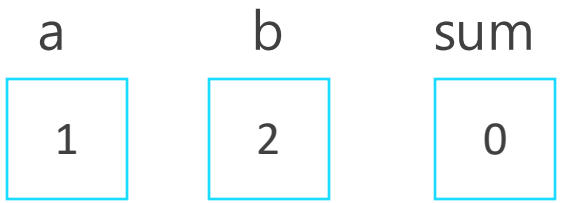
- 시리얼 모니터



단축키: [Ctrl]+[Shift]+M

변수의 사용

- 변수 개념
  - 어떤 값을 저장하기 위한 이름을 가진 공간



- 변수의 선언
  - 변수를 사용하겠다고 컴퓨터에게 알리는 것.
  - 형식

```
자료형 변수명 = 초기값;
```

- 변수의 선언 예

```
// 숫자형
int  a=1;       // 작은 수를 집어 넣을 수 있다.
long b=50000L; // 큰 수를 집어 넣을 수 있다.

// 문자형
char c='A';          // 문자 A를 넣는다.
string d="Hello";  // 문자열 Hello를 넣는다.

// 부울형
bool e=true;  // true, false 만 가능
```

- 동일 값

  HIGH == 1 == true == TRUE

  LOW == 0 == false == FALSE

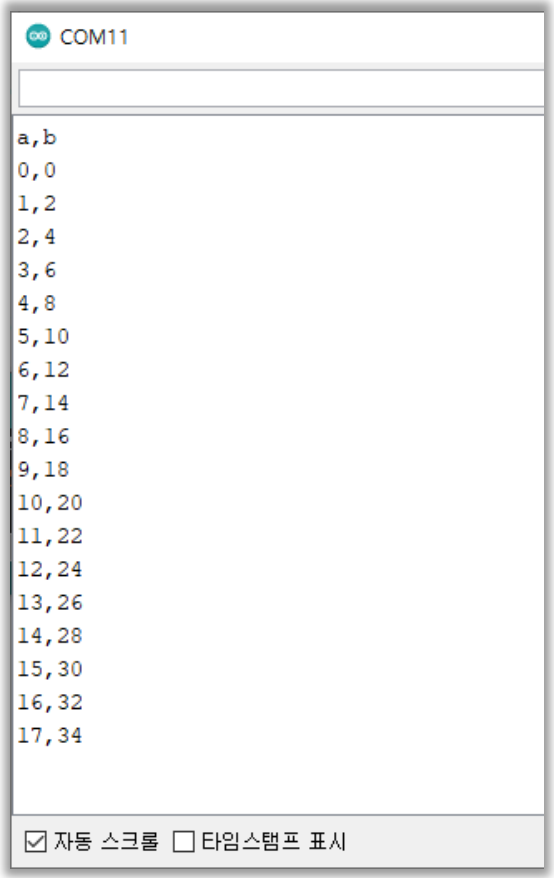
Serial 포트를 출력창으로 사용하기

- 소스코드2

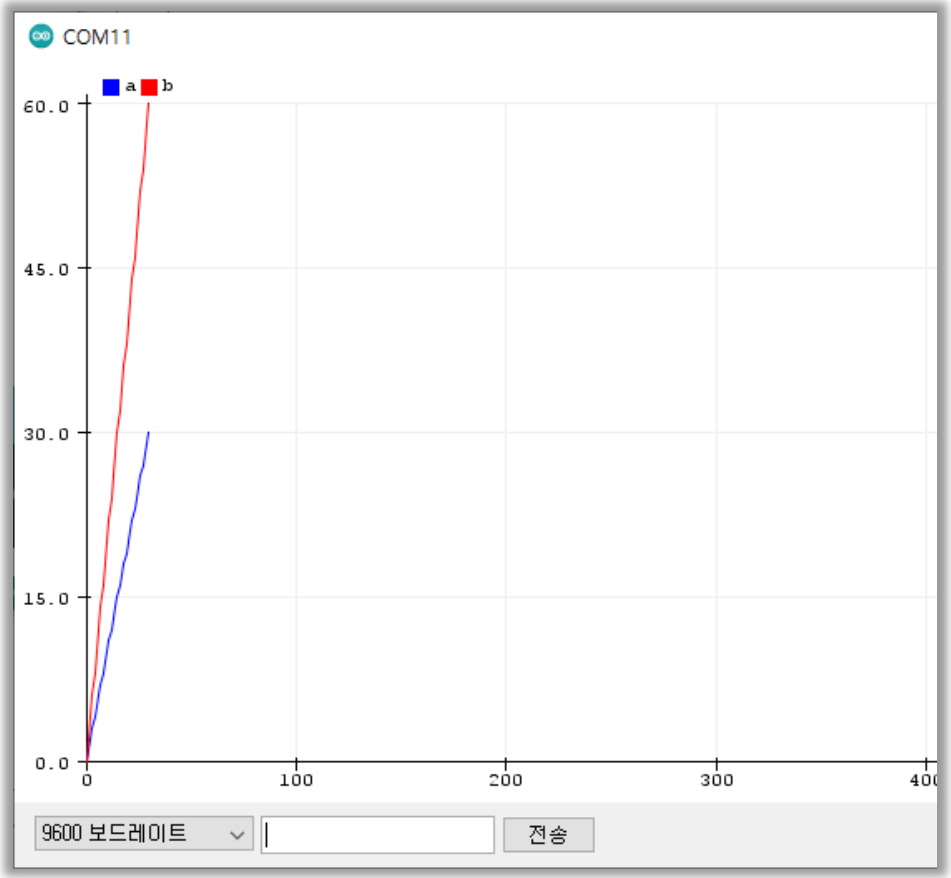
1-6.ino

```
int a, b;

void setup() {
  Serial.begin(9600);
  Serial.println("a,b");
}

void loop() {
  char str[50];
  sprintf(str, "%d,%d", a,b);
  Serial.println(str);

  a=a+1;
  b=b+2;
  delay(500);
}
```

- 시리얼 모니터



- 시리얼 플로터



단축키: [Ctrl]+[Shift]+L

제어문

score

90

- if 문

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

void loop() {
  int score = 90;

  if(score == 100)
    Serial.println("Very Good!");

  if(score >= 60) {
    Serial.println("You Passed");
  }

  delay(10000);
}
```

- if … else 문

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

void loop() {
  int score = 90;

  if(score >= 60) {
    Serial.println ("You Passed");
  }
  else {
    Serial.println("You Failed");
  }

  delay(10000);
}
```

반복문

- for 반복문

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

void loop() {
  int i, sum = 0;

  for(i=1; i<=10; i++) {
    sum = sum + i;

    Serial.print(i);
    Serial.print(" : ");
    Serial.println(sum);
  }
  delay(10000);
}
```

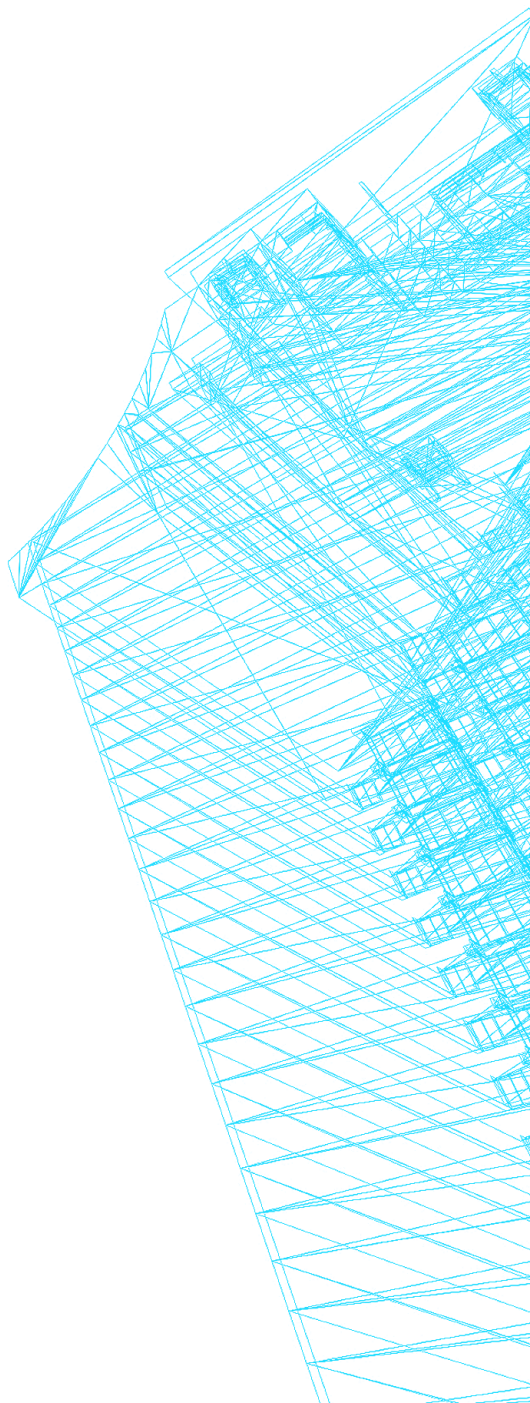
- while 반복문

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

void loop() {
  int i=1, sum = 0;

  while(i<=10) {
    sum = sum + i;

    Serial.print(i);
    Serial.print(" : ");
    Serial.println(sum);
    i++;
  }
  delay(10000);
}
```

sum	i
1	1

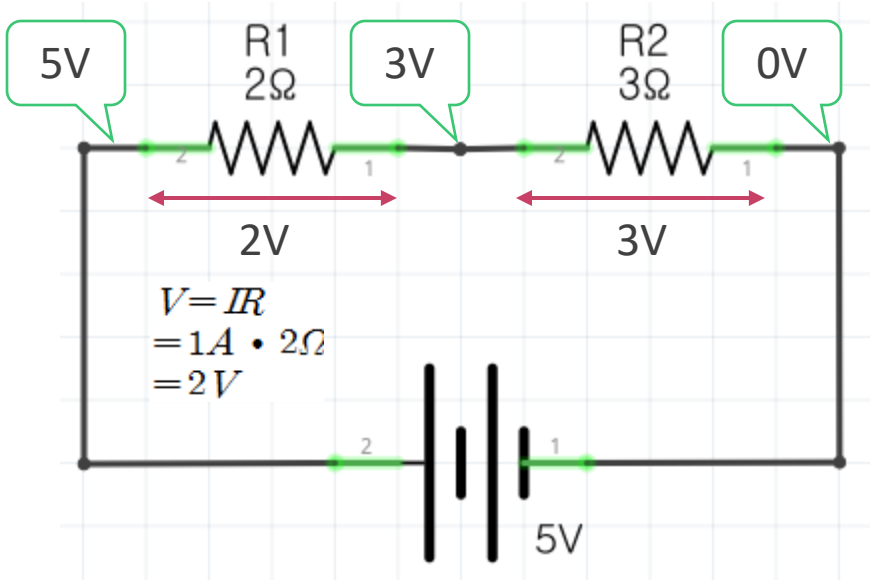
Digital Input

- 선수학습 - 전압강하
- 버튼회로 설계 하기
  - Active HIGH 버튼 만들기 (PULL-DOWN 저항)
  - Active LOW 버튼 만들기 (PULL-UP 저항)
  - INPUT_PULLUP
- 토글 버튼 구현

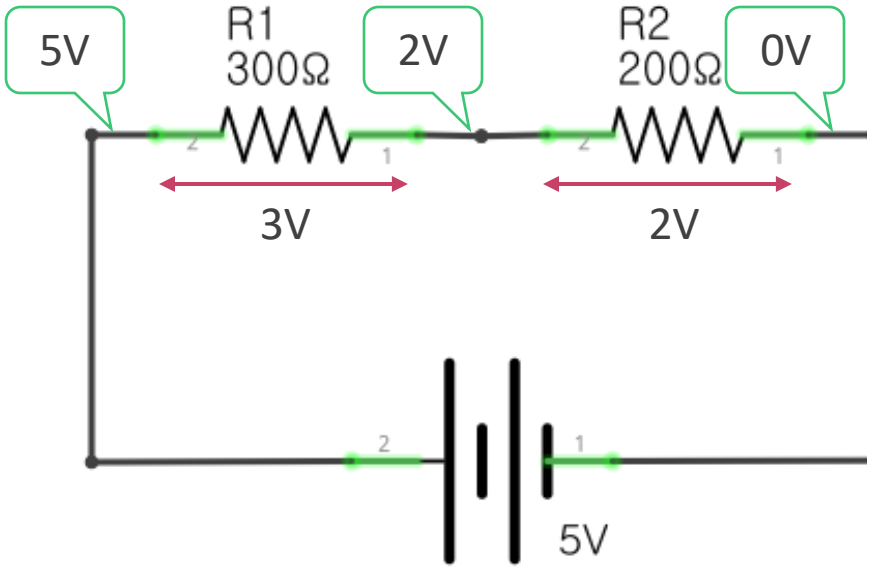


선수학습 - 전압강하

- 전류가 두 전위 사이를 흐를 때 저항을 직렬로 여러 개 연결하면 전류가 각 저항을 통과할 때마다 옴의 법칙[전압(V)= 전류(I)·저항(R)]만큼 전압이 작아져 나타나는 현상을 전압강하라 한다. 이때 전체 전압은 각각에 걸린 전압의 합이 된다.



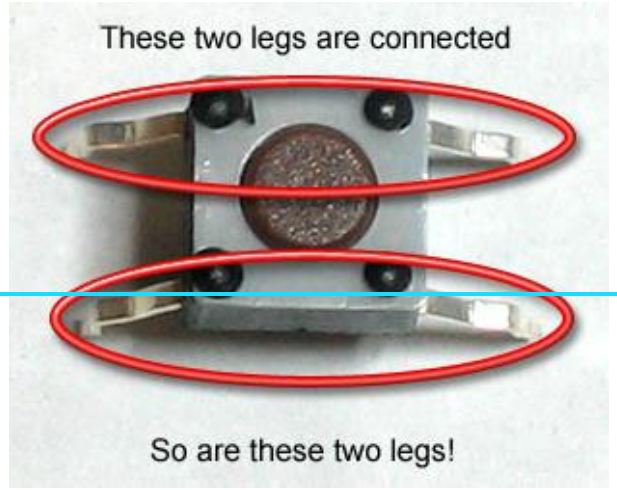
$$I = \frac{V}{R} = \frac{5}{2+3} = 1A$$

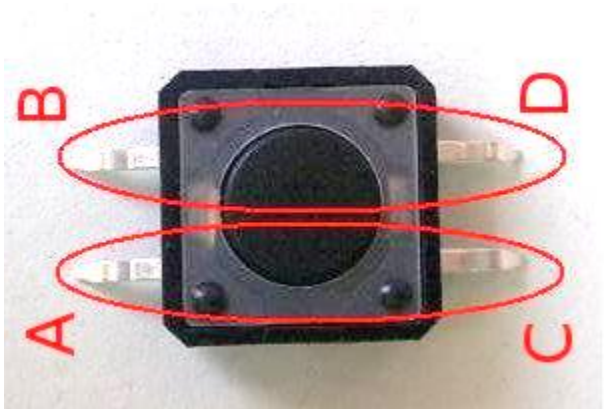


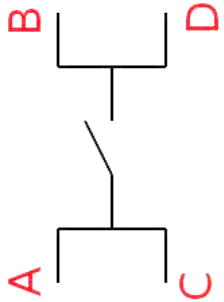
퀴즈) 만약 선이 중간이 끊어졌다면?

버튼

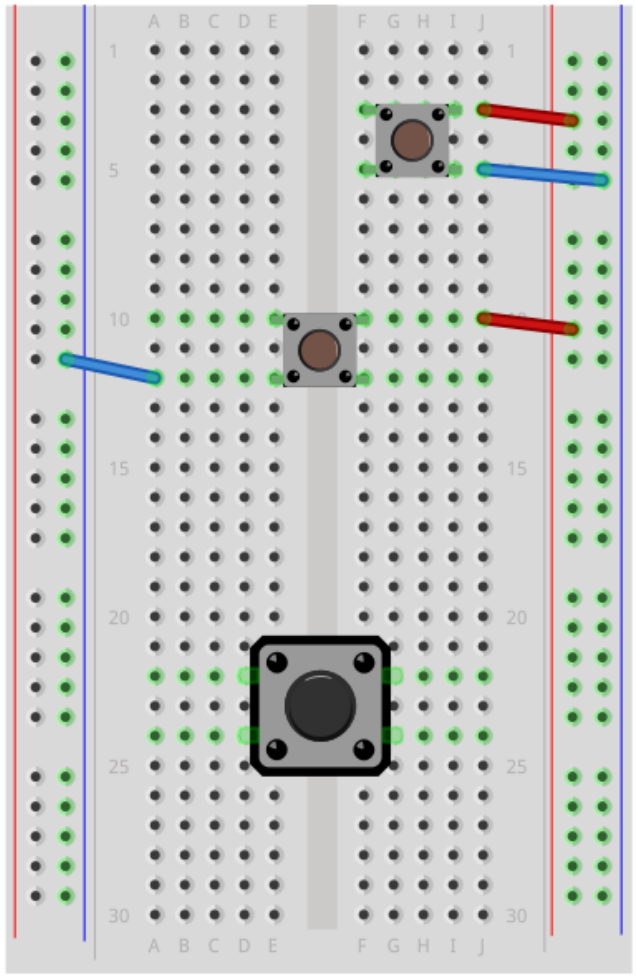
- 푸시 버튼







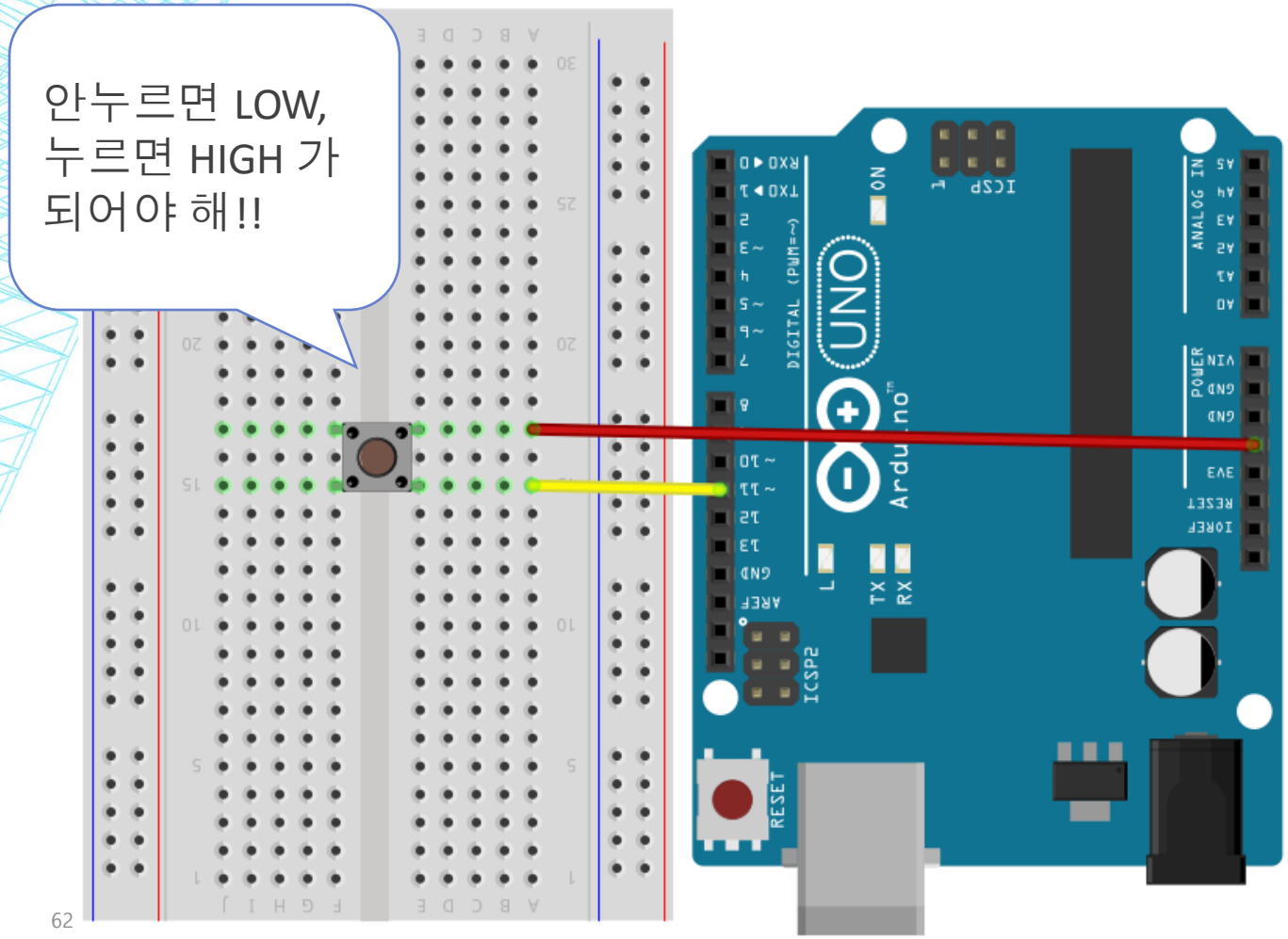
- 브레드보드에 배치 예시



누르면 1(=5V, =HIGH)이 되는

Active HIGH 버튼 회로 설계

- 첫 번째 시도



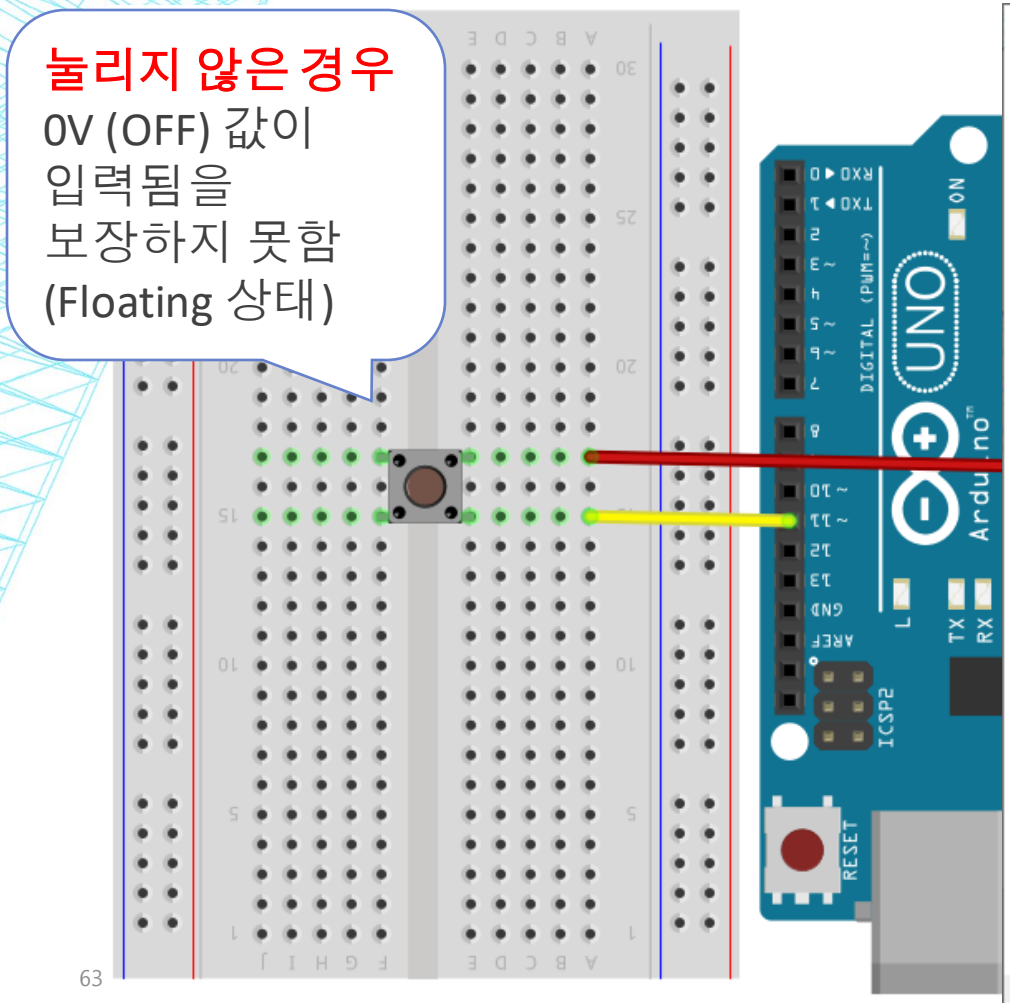
- 소스코드

2-1.ino

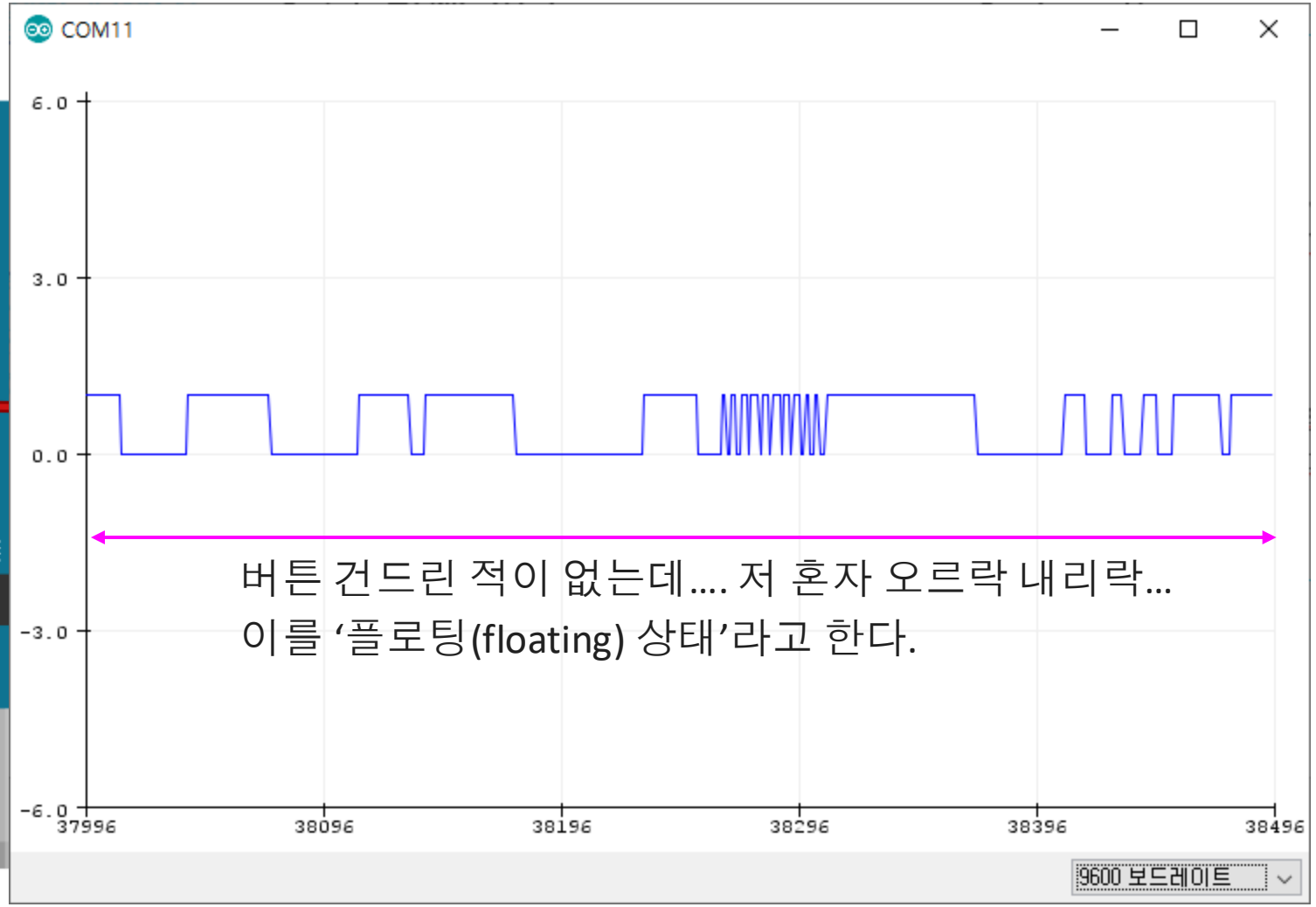
```
#define BTN 11

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

void loop() {
  bool r = digitalRead(BTN);
  Serial.println(r);
  delay(100);
}
```

Active HIGH 버튼 회로 설계

- 첫 번째 시도



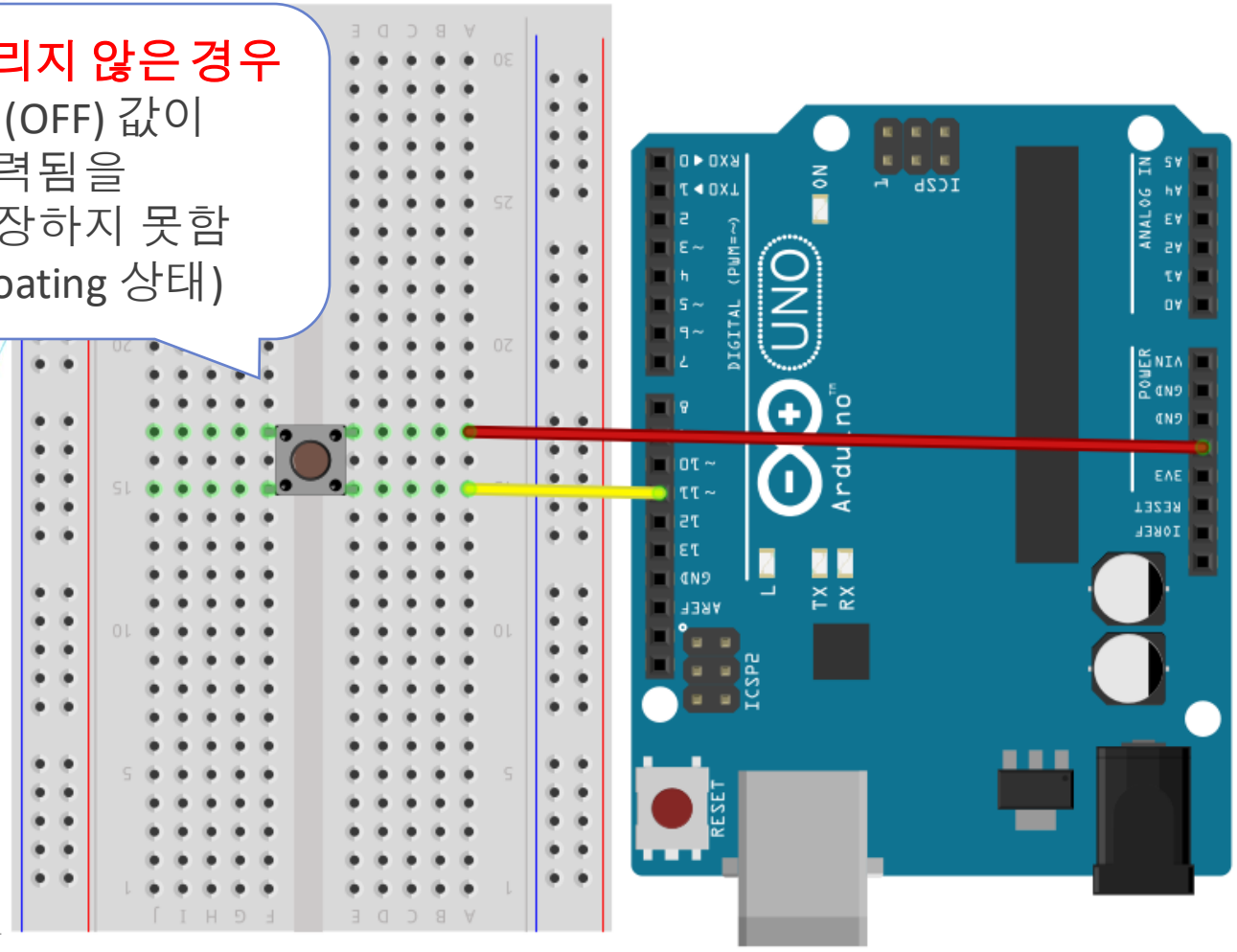
- 실험결과



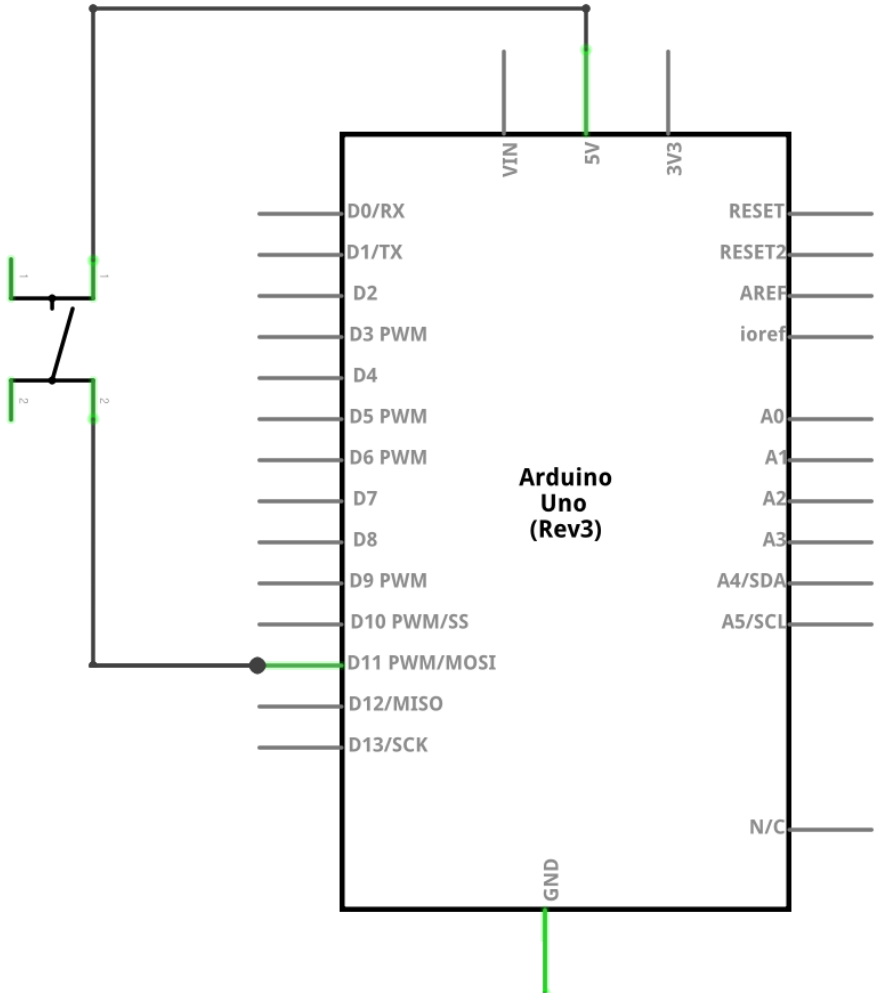
Active HIGH 버튼 회로 설계

- 첫 번째 시도

눌리지 않은 경우
0V (OFF) 값이
입력됨을
보장하지 못함
(Floating 상태)



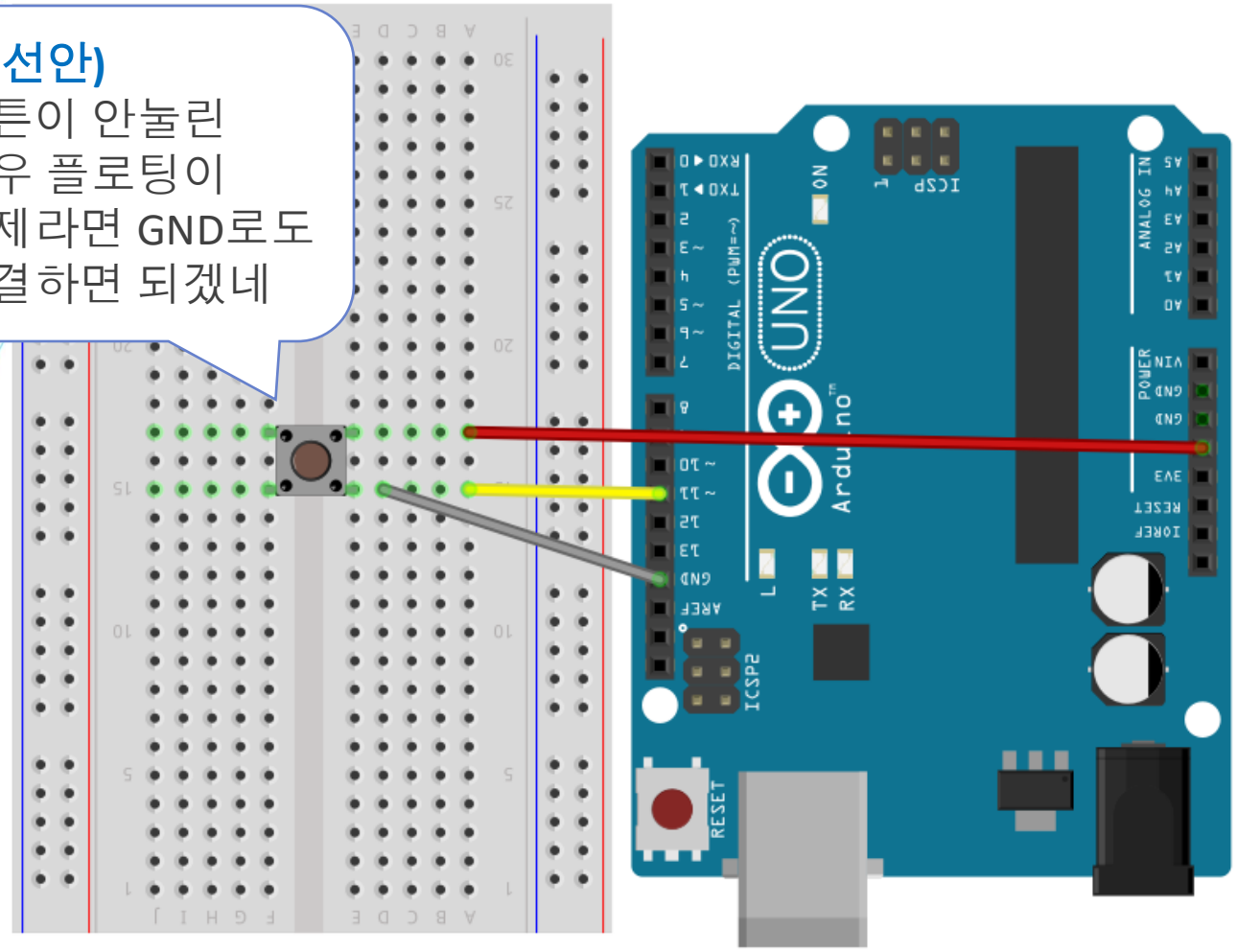
- 회로도



버튼 회로 설계(Active HIGH)

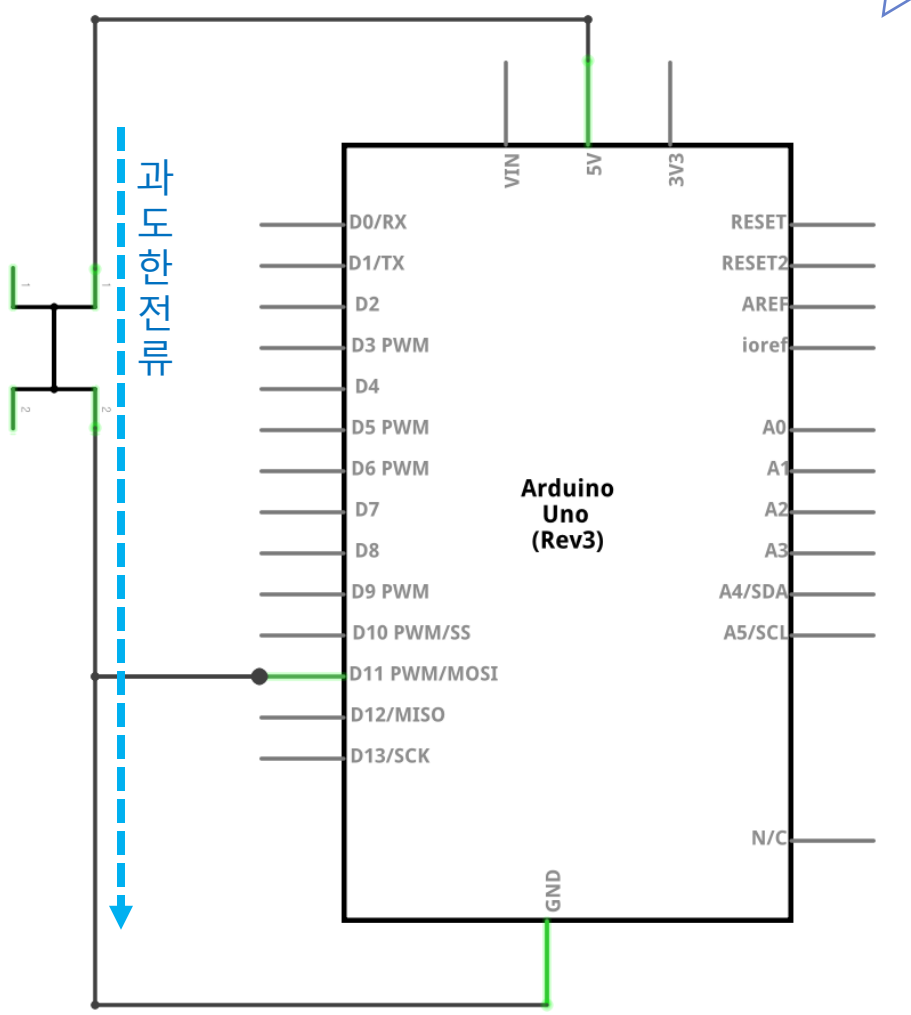
- 두 번째 시도

(개선안)
버튼이 안눌린 경우 플로팅이 문제라면 GND로도 연결하면 되겠네



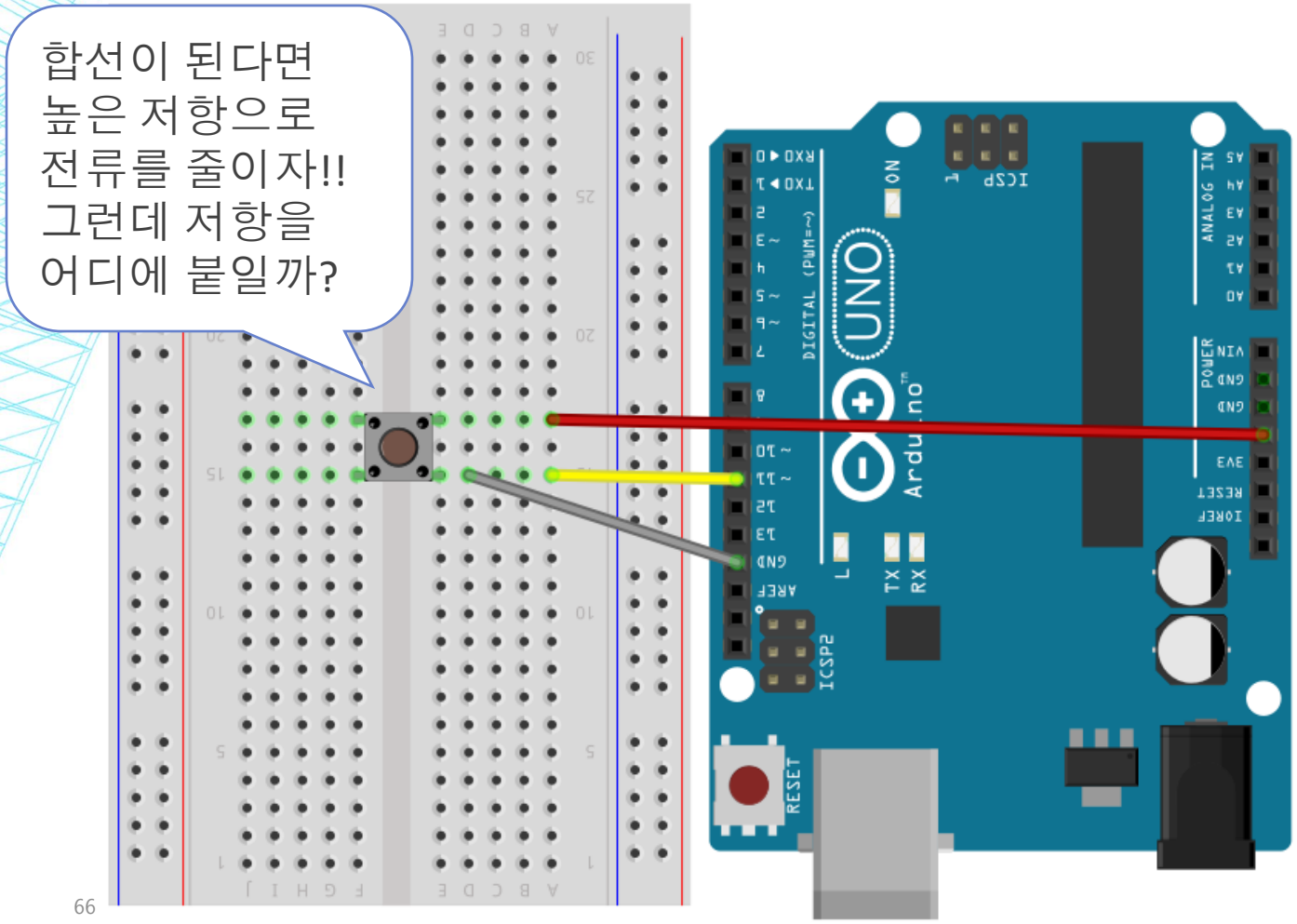
- 회로도

엥 안되네?

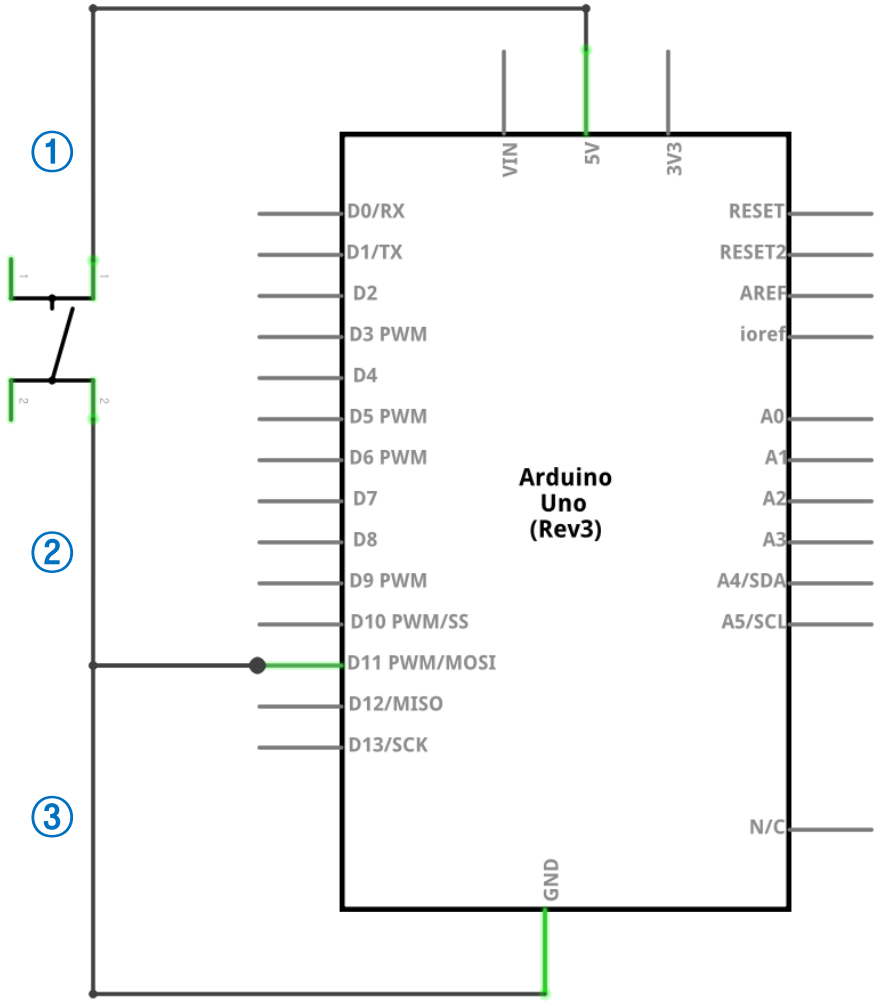


Active HIGH 버튼 회로 설계

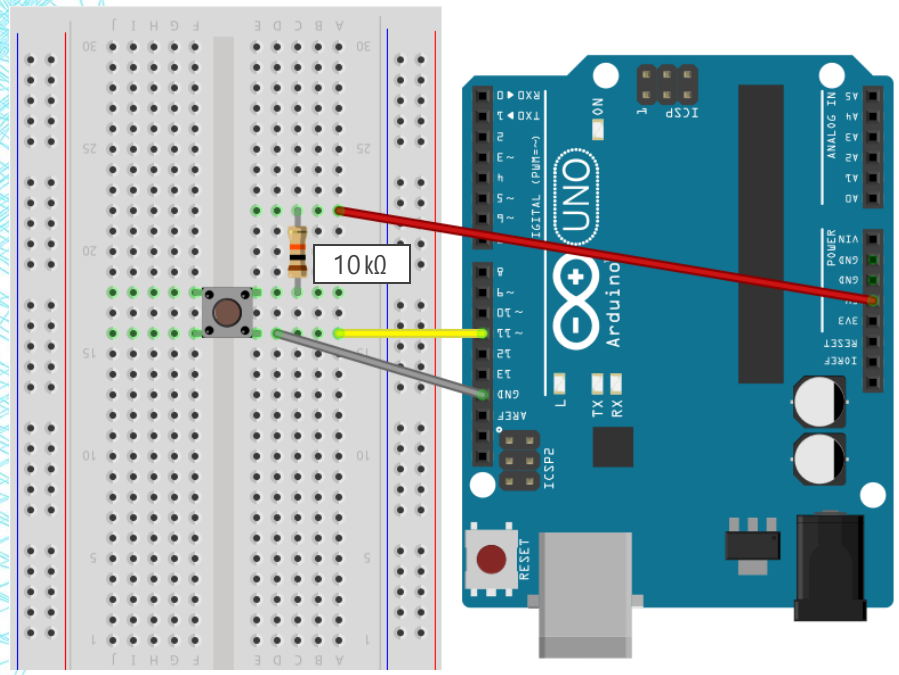
- 세 번째 시도



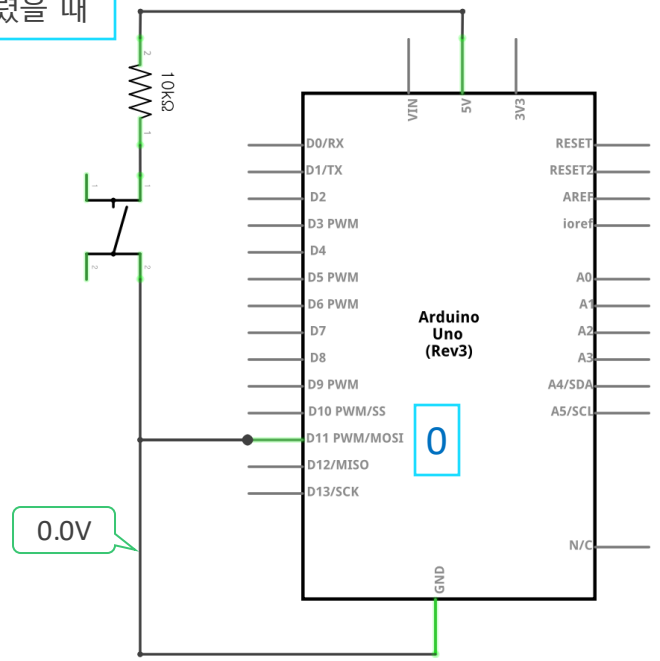
- 회로도



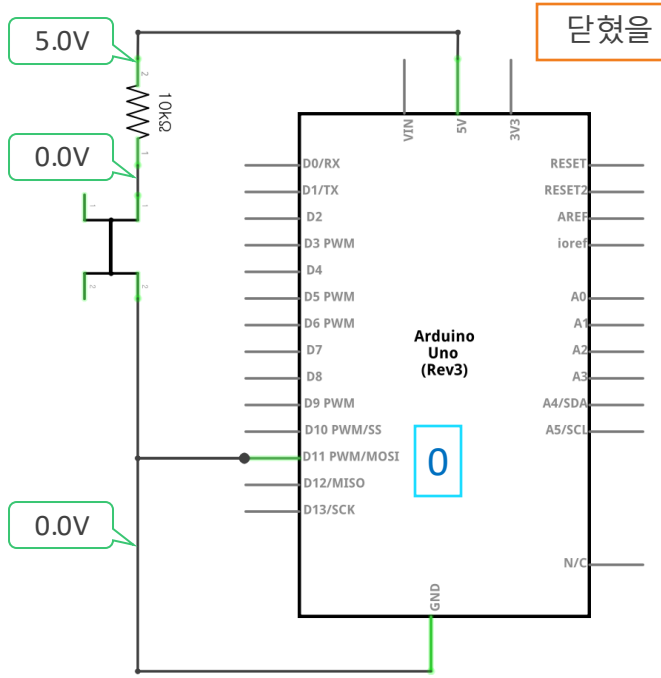
①



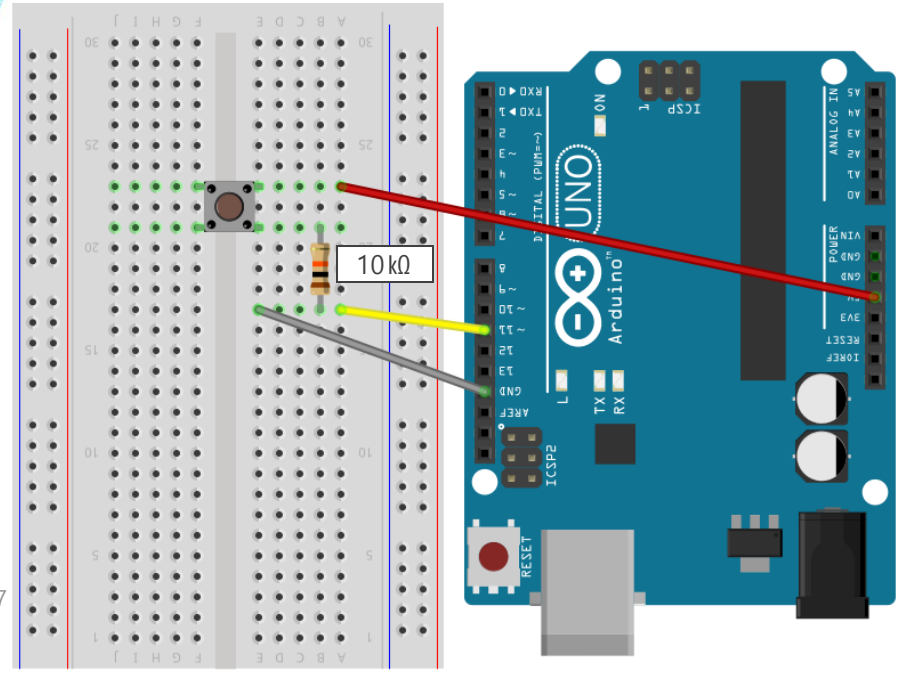
열렸을 때



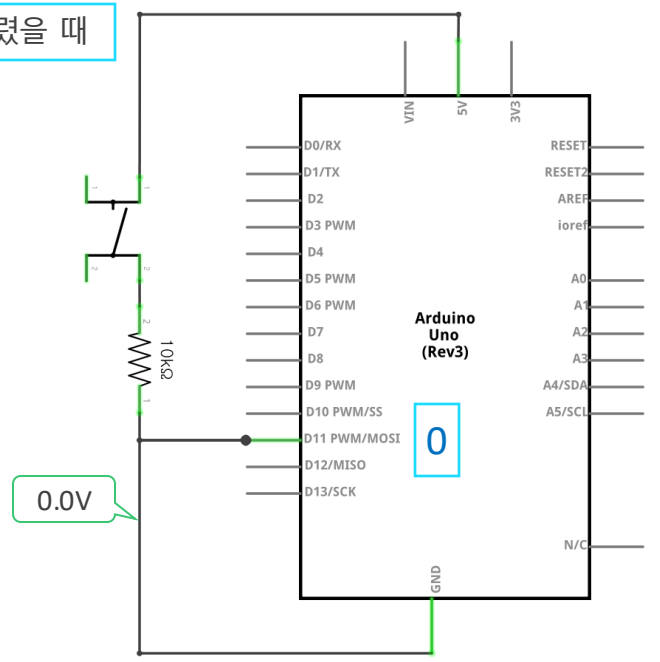
닫혔을 때



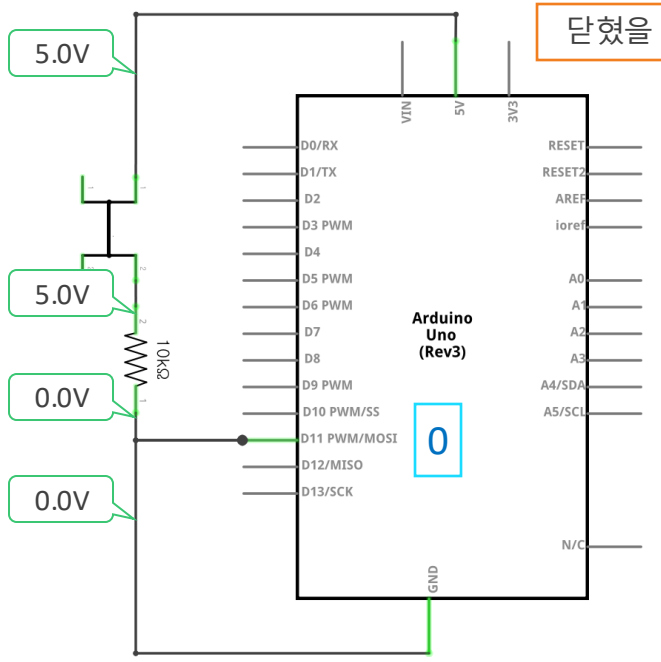
②



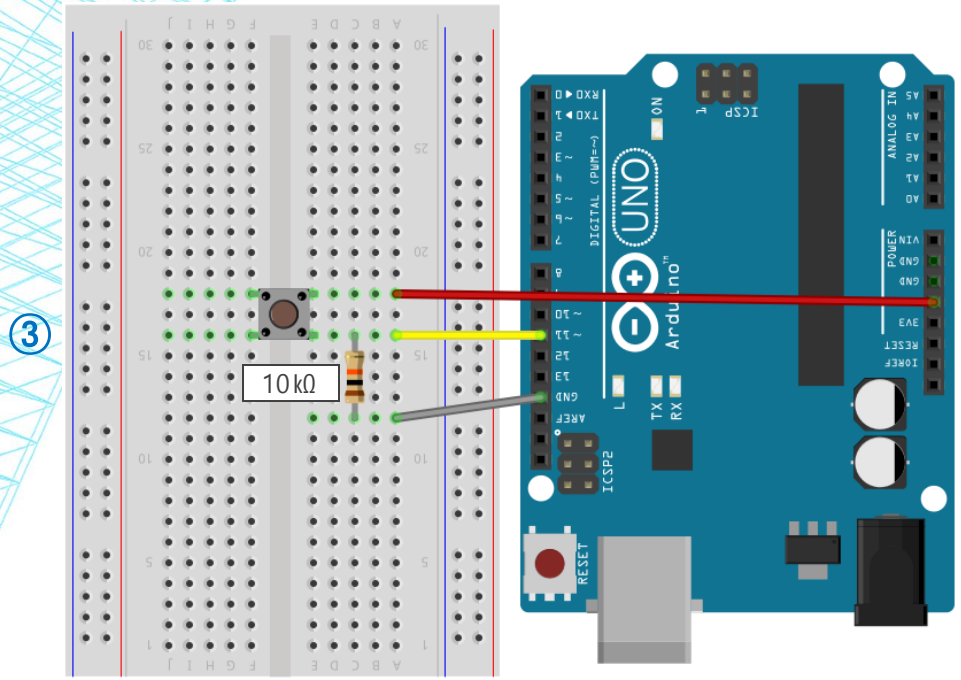
열렸을 때

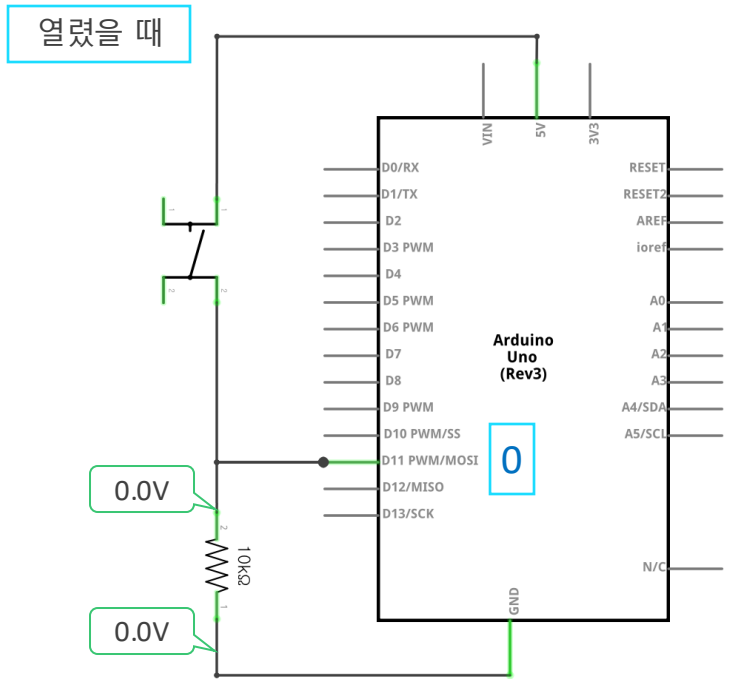


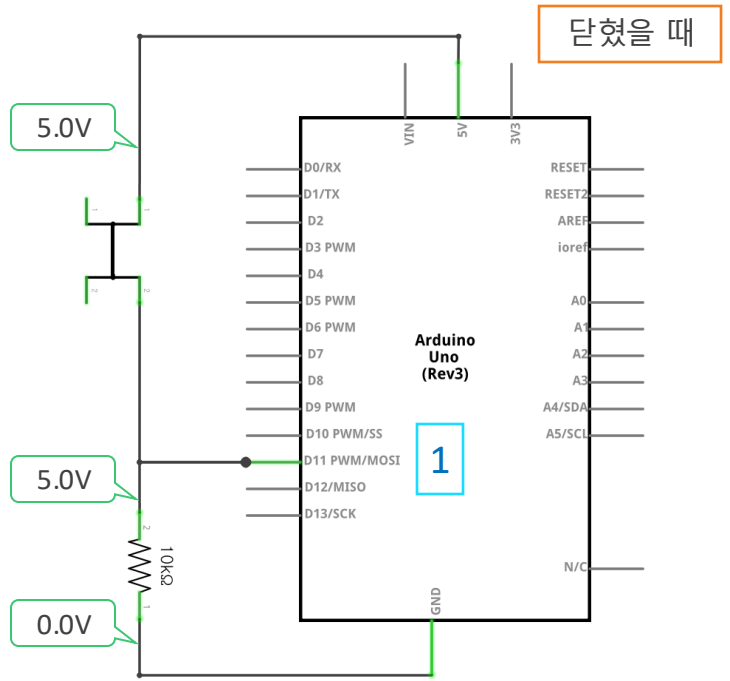
닫혔을 때



Active HIGH 버튼 회로 설계





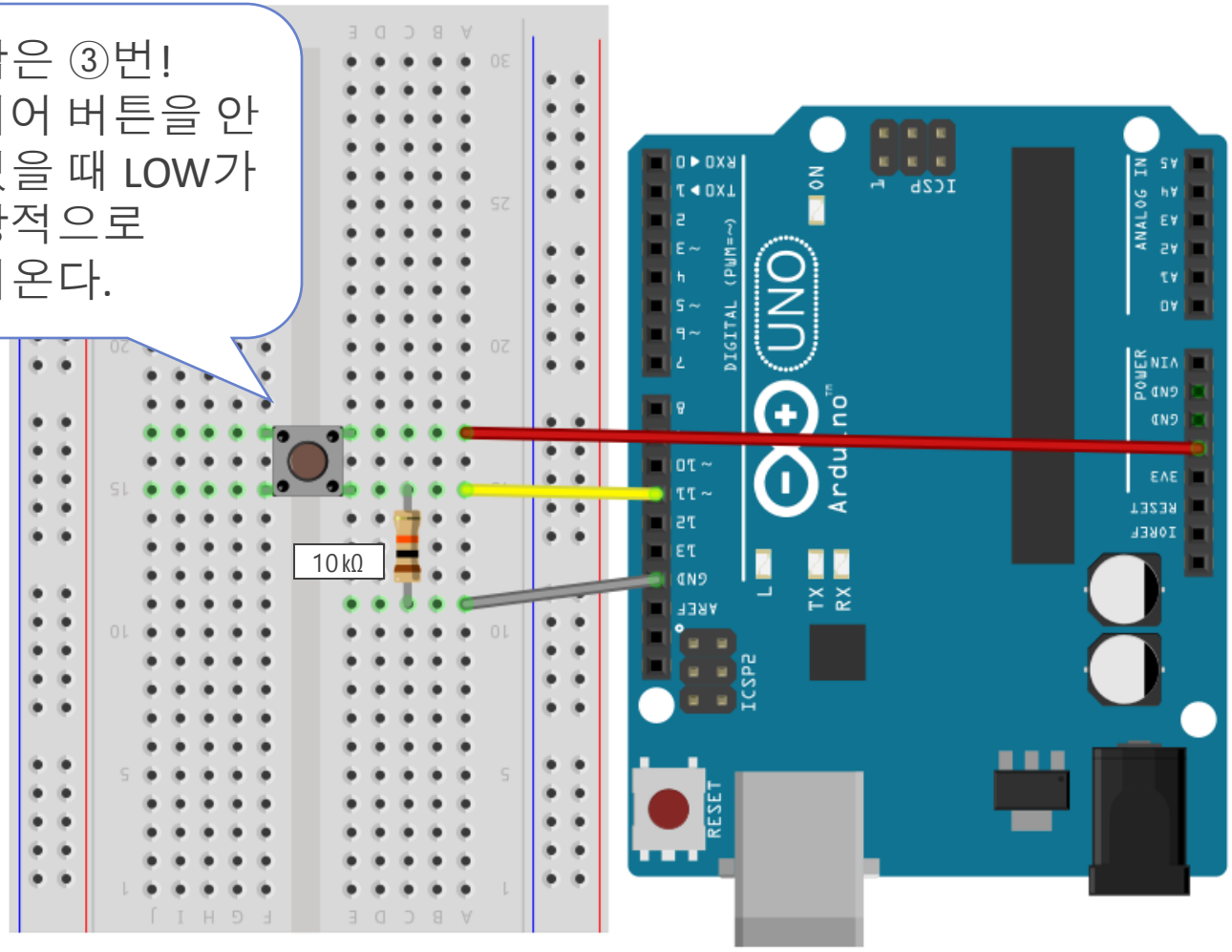


- 풀-다운 저항(기본 값을 LOW로 만드는 저항) 사용
  - 스위치가 열려 있을 때, 플로팅 상태를 해결하고 LOW(0V)로 끌어내리기 위해 GND쪽에 첨부하는 저항

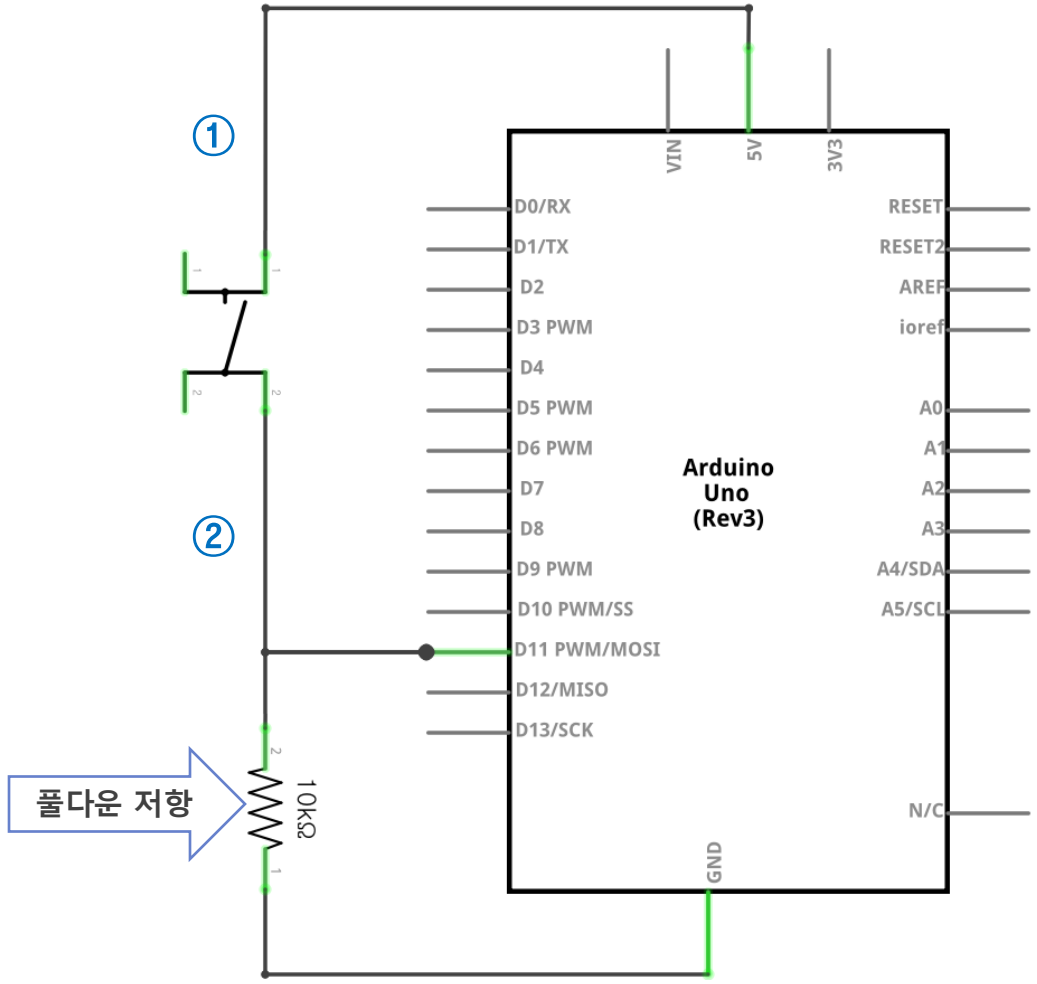
Active HIGH 버튼 회로 설계

- 세 번째 시도 – 풀다운 저항

정답은 ③번!
드디어 버튼을 안 눌렀을 때 LOW가 정상적으로 들어온다.



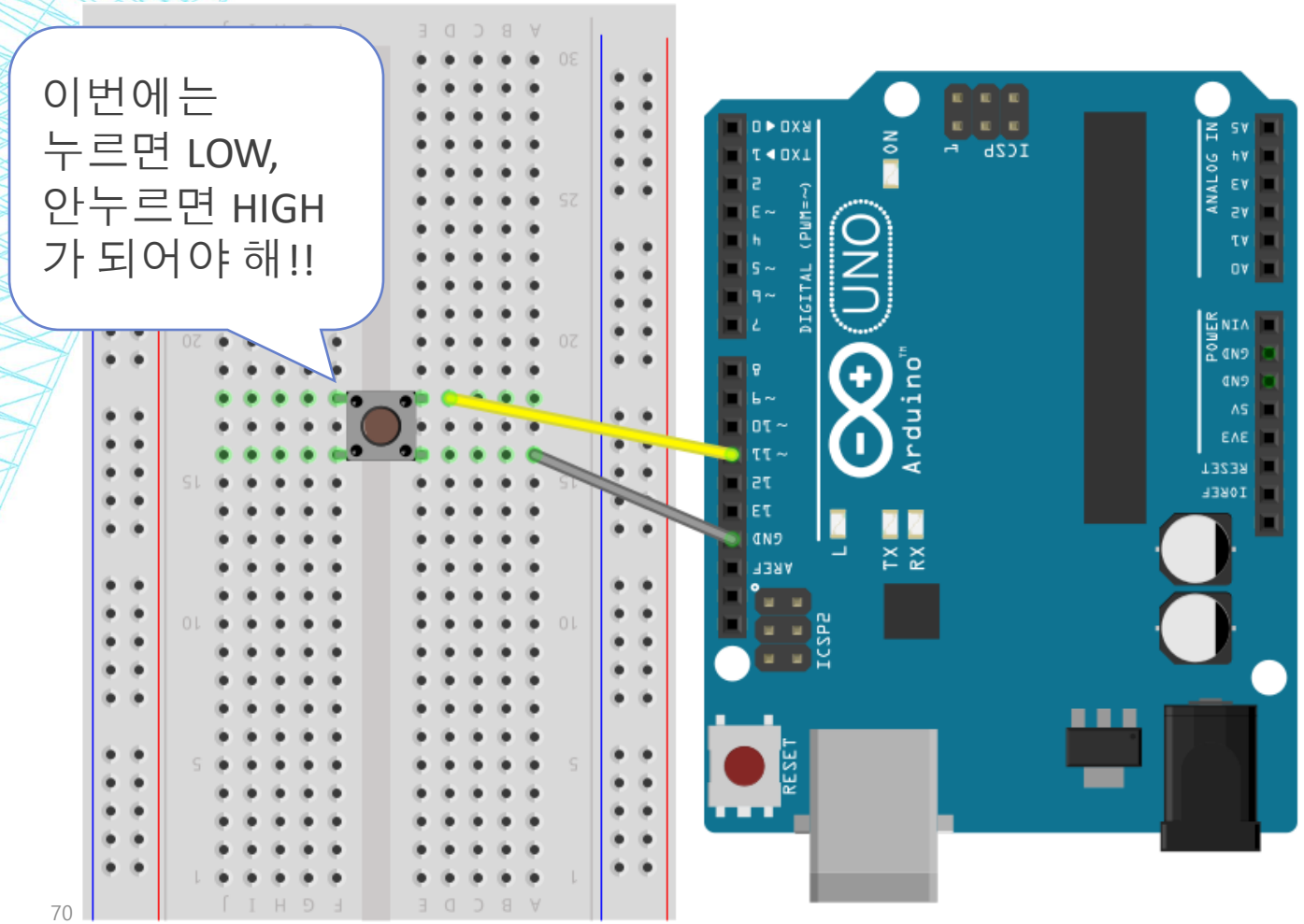
- 회로도



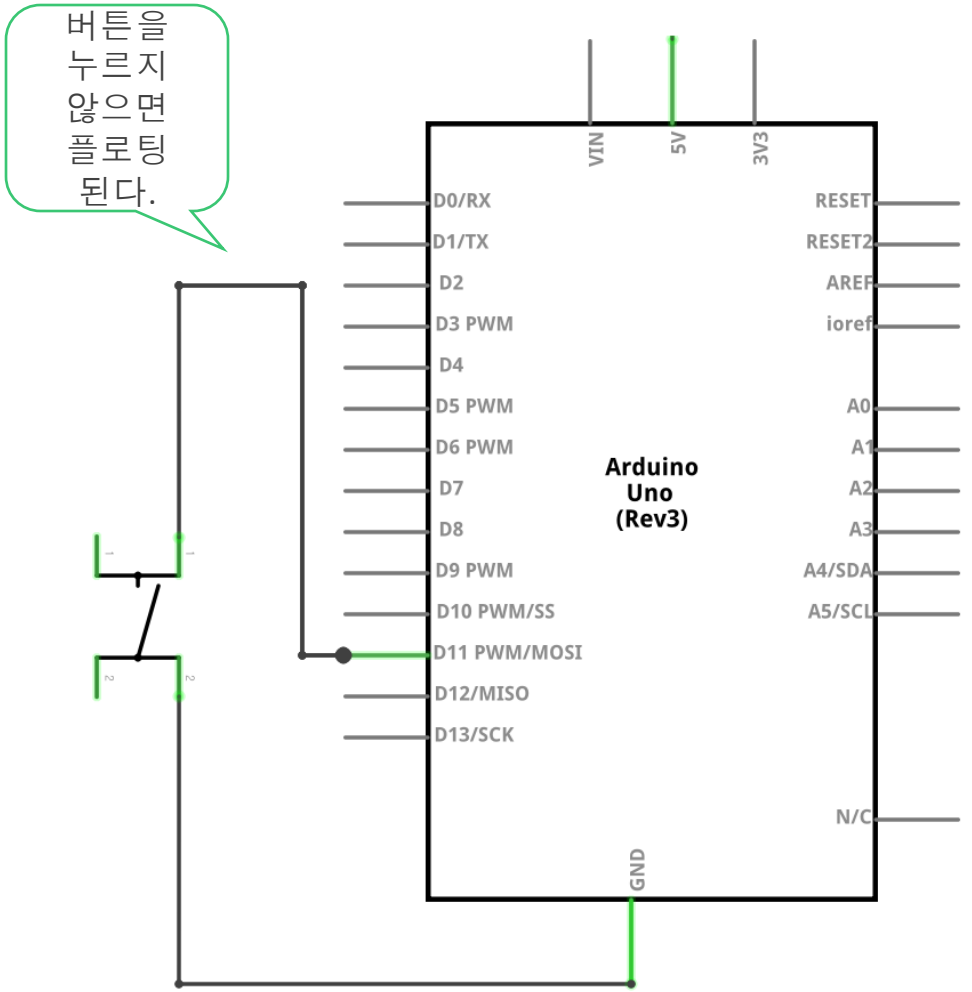
누르면 0(=0V, =LOW)이 되는

Active LOW 버튼 회로 설계

- 첫 번째 시도



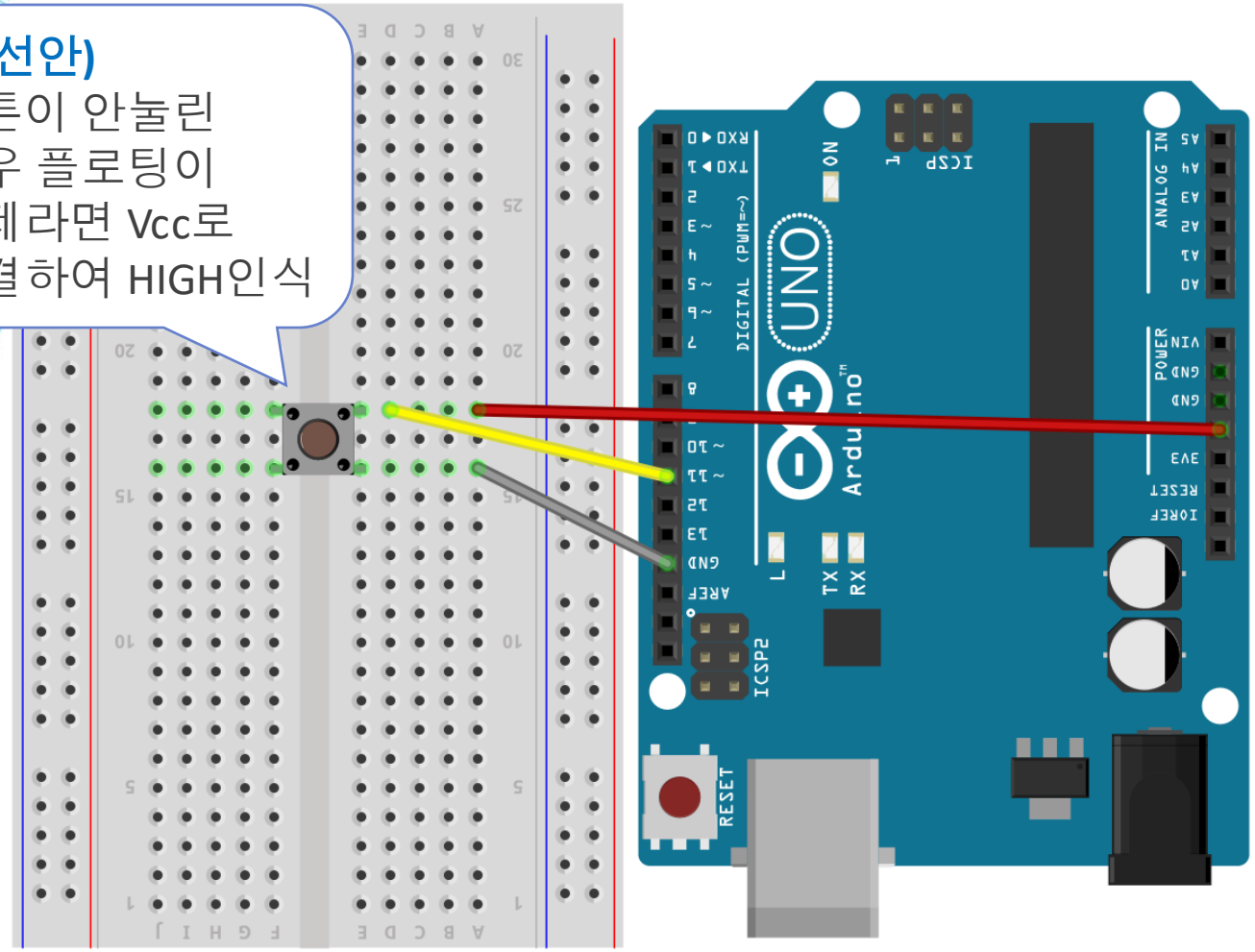
- 회로도



Active LOW 버튼 회로 설계

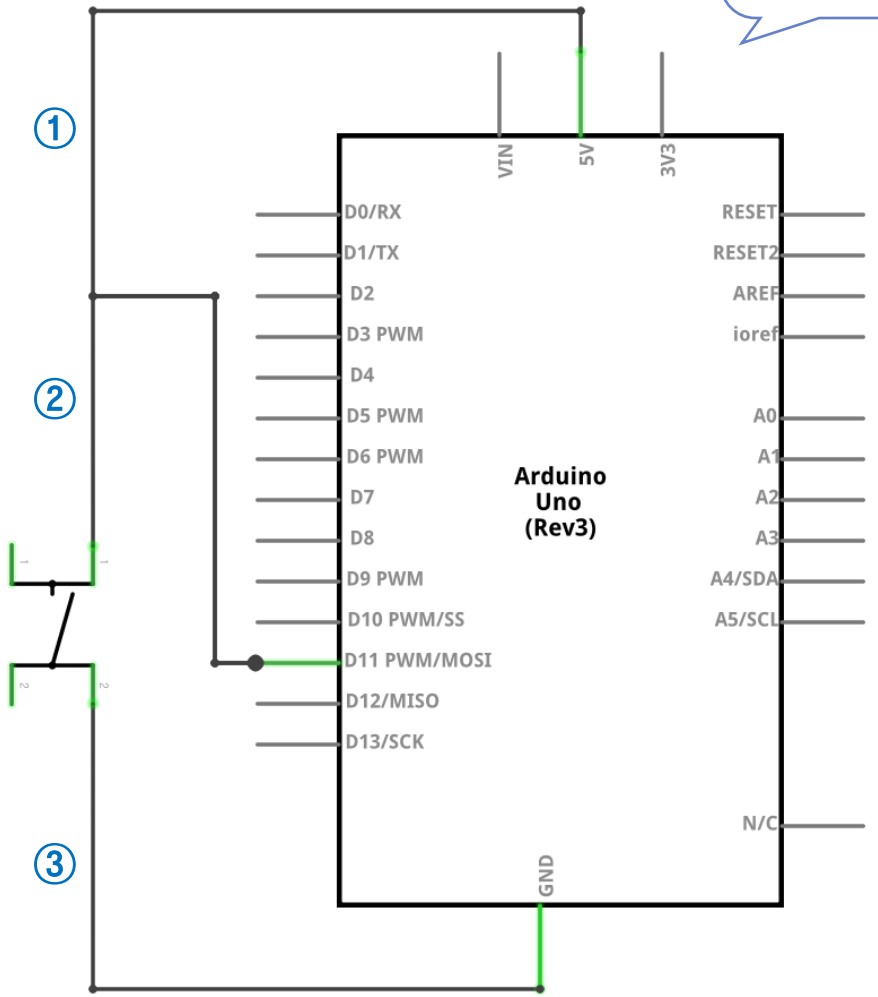
- 두 번째 시도

(개선안)
버튼이 안눌린 경우 플로팅이 문제라면 Vcc로 연결하여 HIGH인식

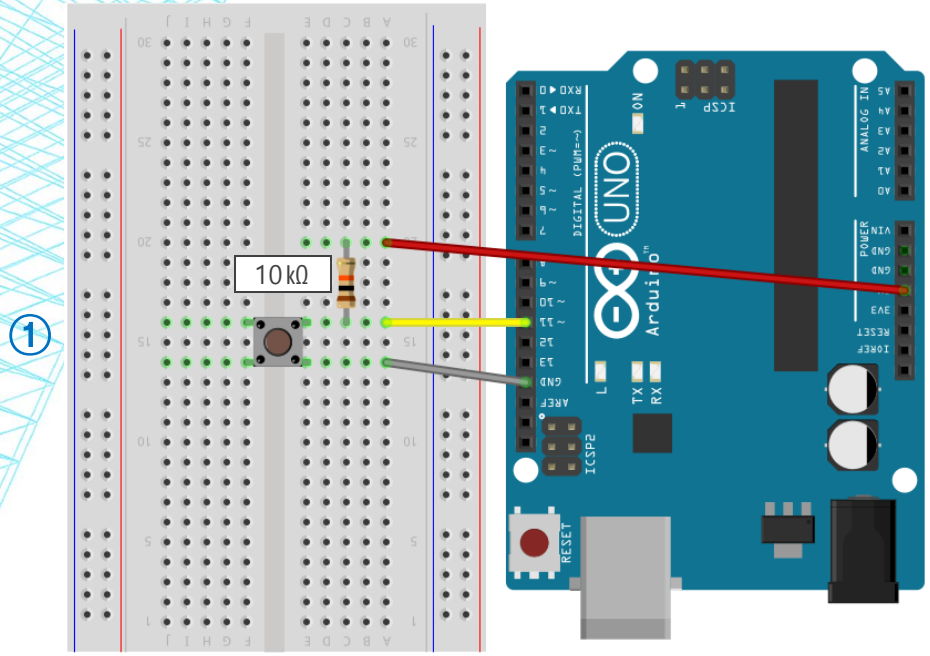


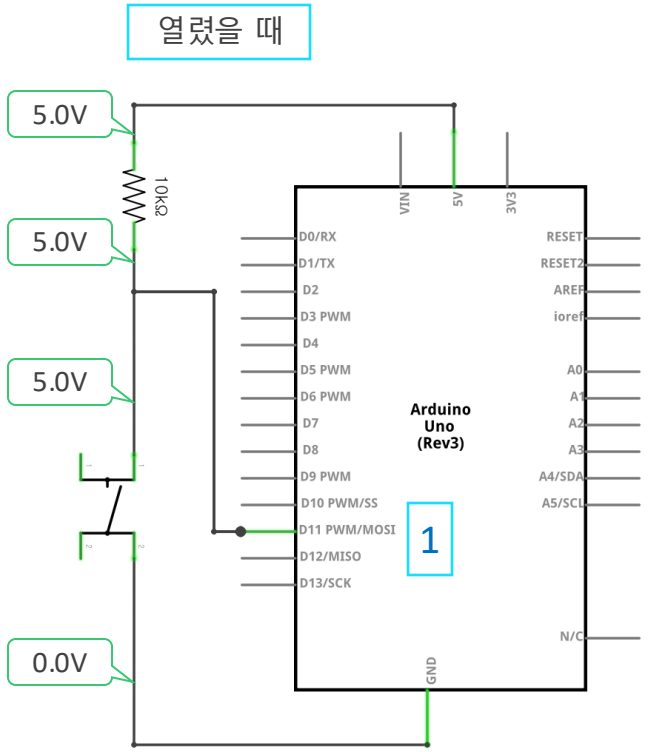
- 회로도

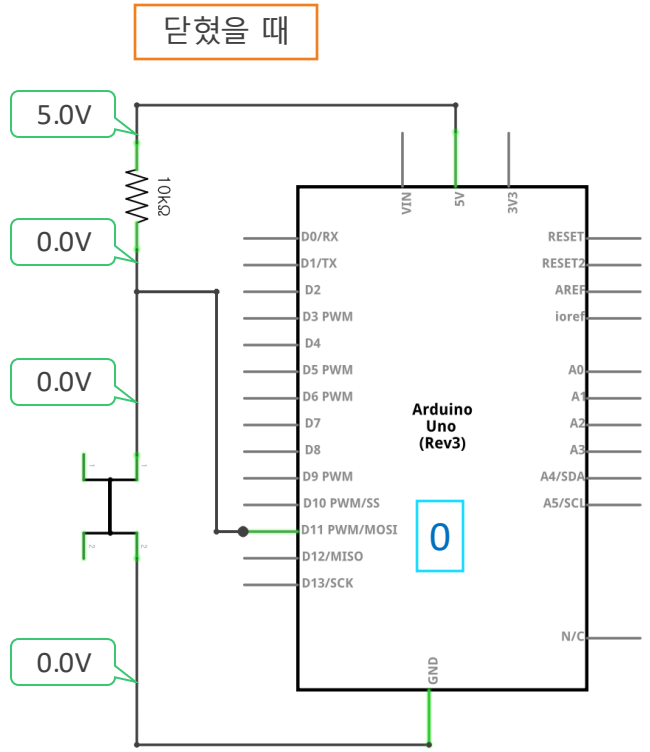
저항을 어디에 붙일까?



Active LOW 버튼 회로 설계

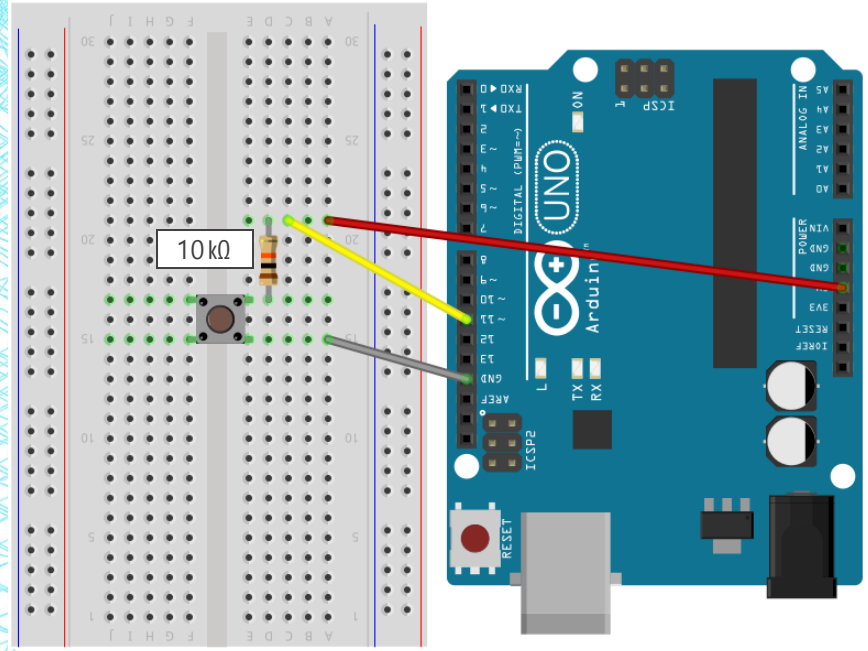




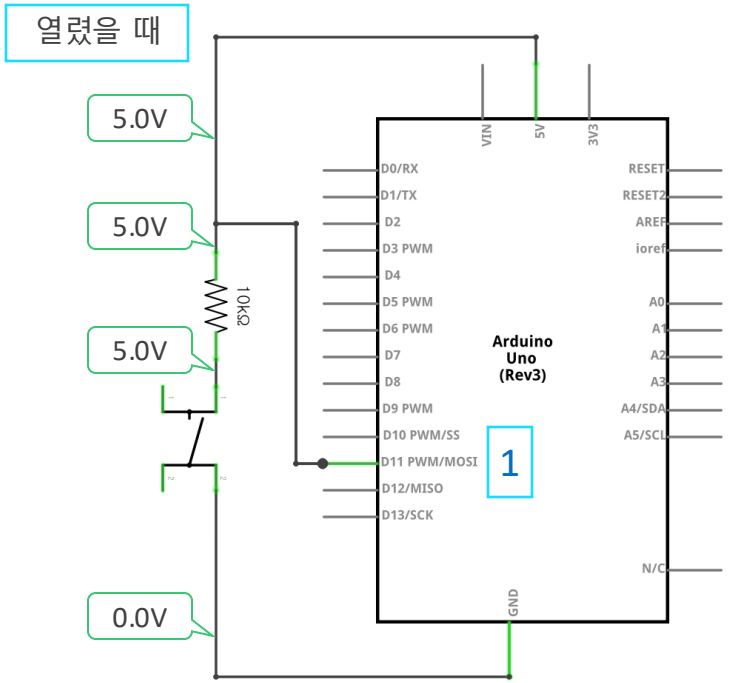


- 풀-업 저항(기본 값을 HIGH로 만드는 저항) 사용
  - 스위치가 열려 있을 때 플로팅 상태를 해결하고, 전압을 HIGH(5V)로 끌어 올리기 위해 Vcc에 첨부하는 저항

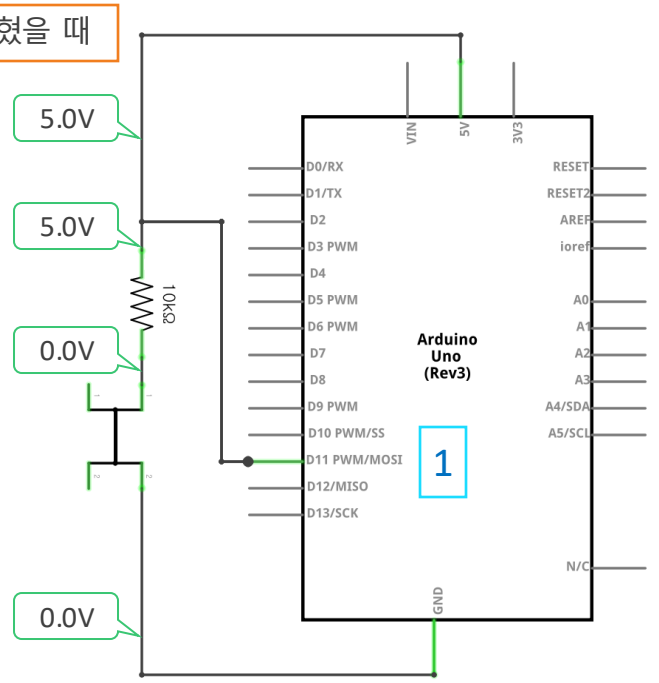
②



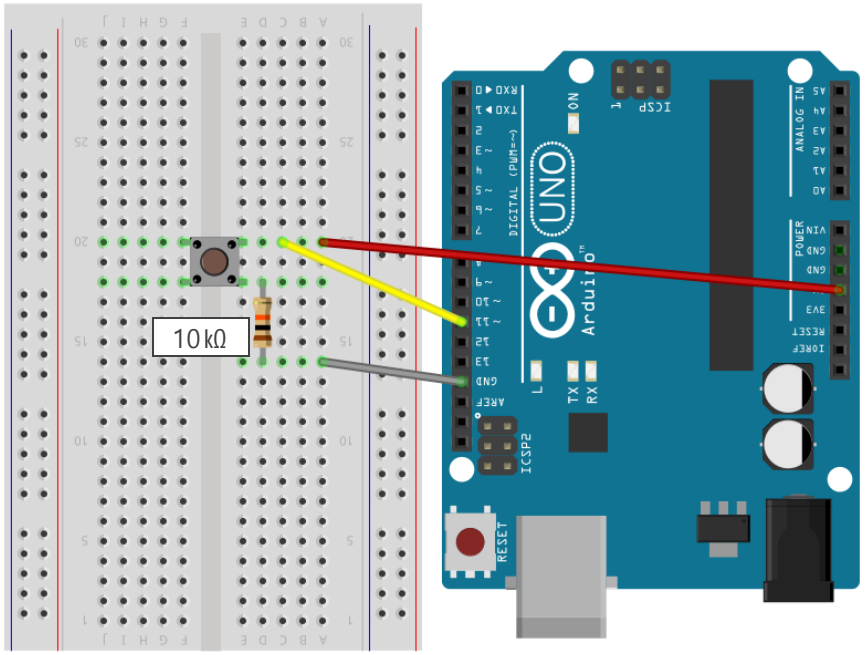
열렸을 때



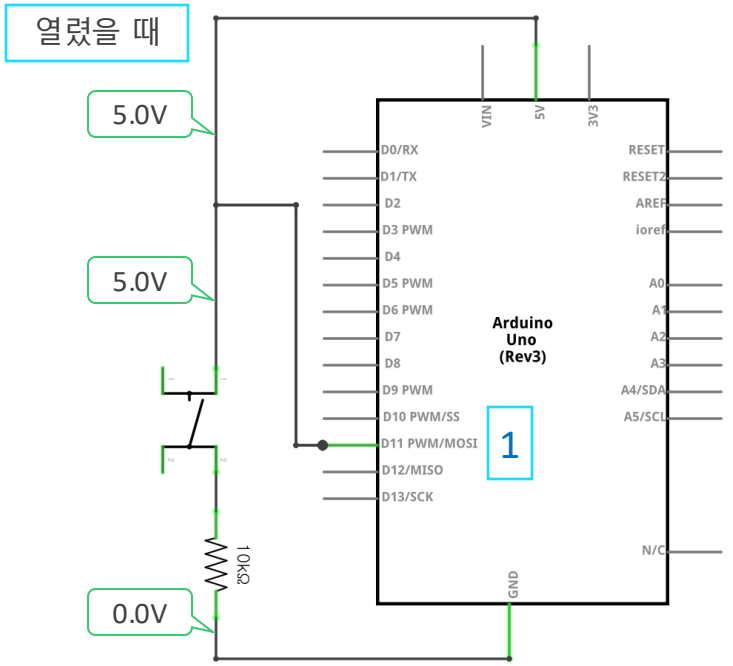
닫혔을 때



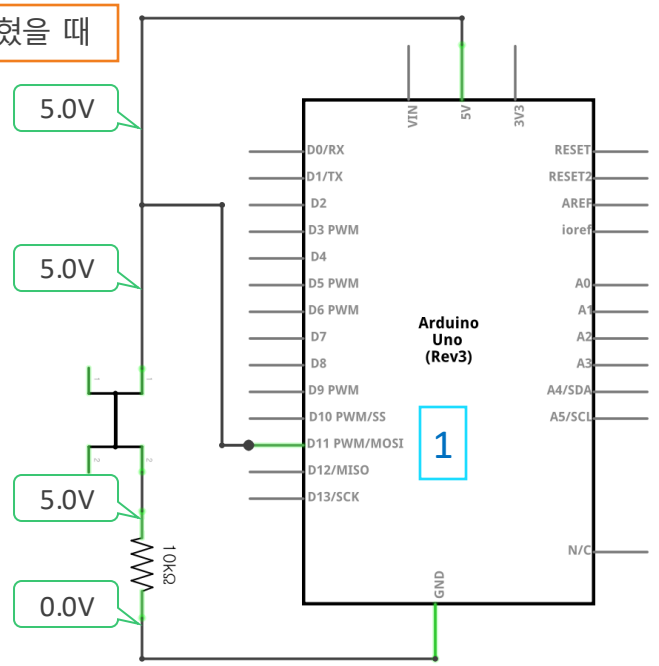
③



열렸을 때



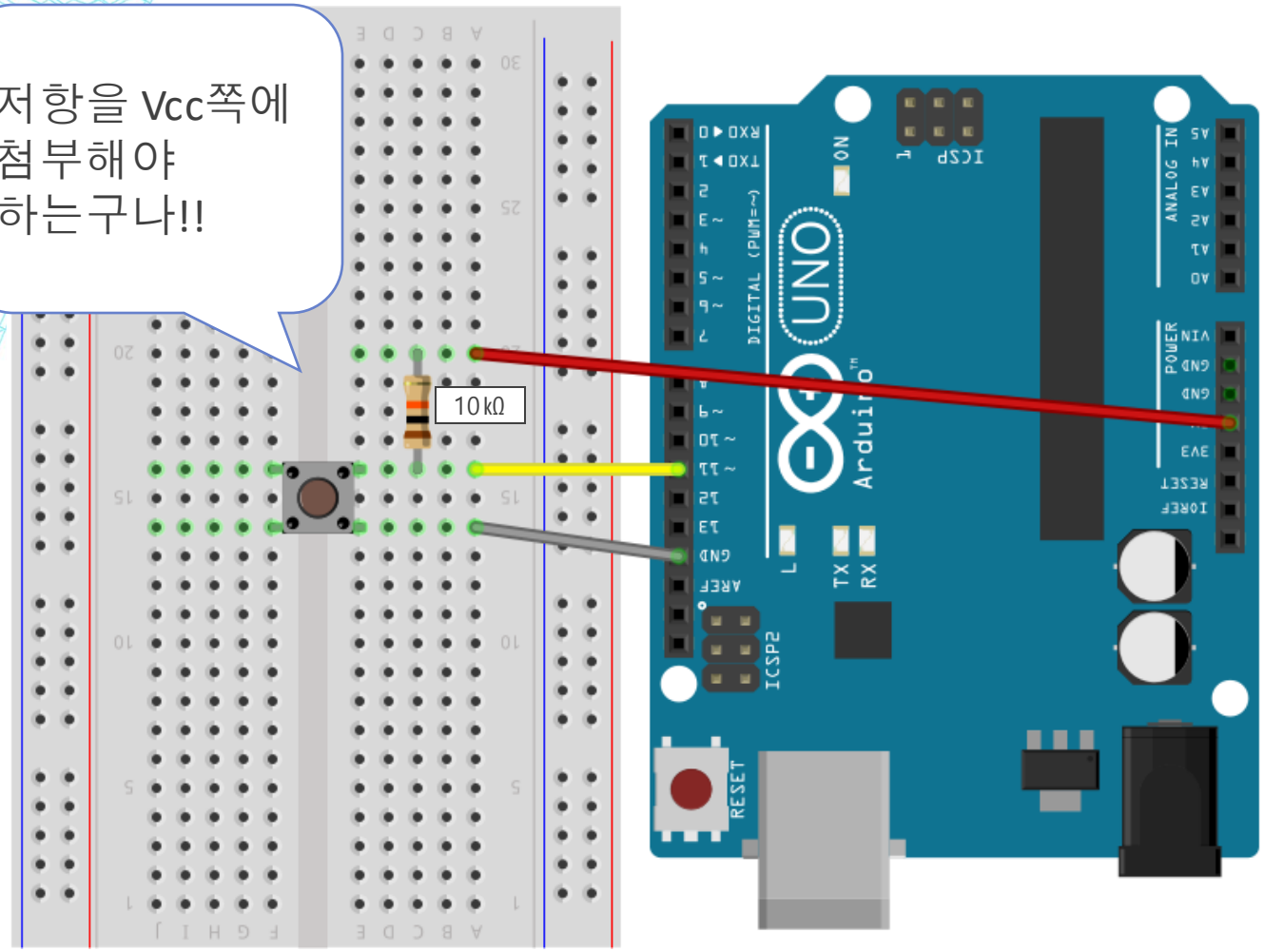
닫혔을 때



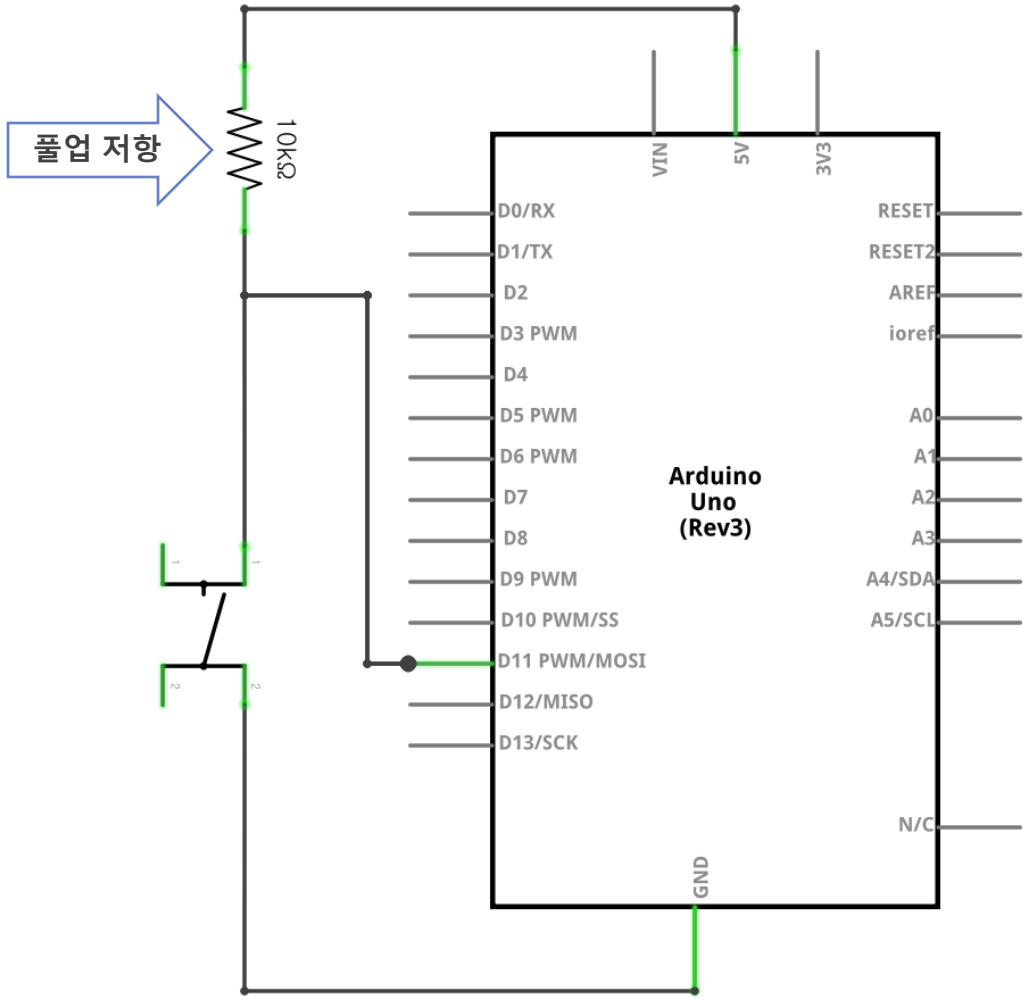
Active LOW 버튼 회로 설계

- 세 번째 시도 – 풀업 저항

저항을 Vcc쪽에 첨부해야 하는구나!!

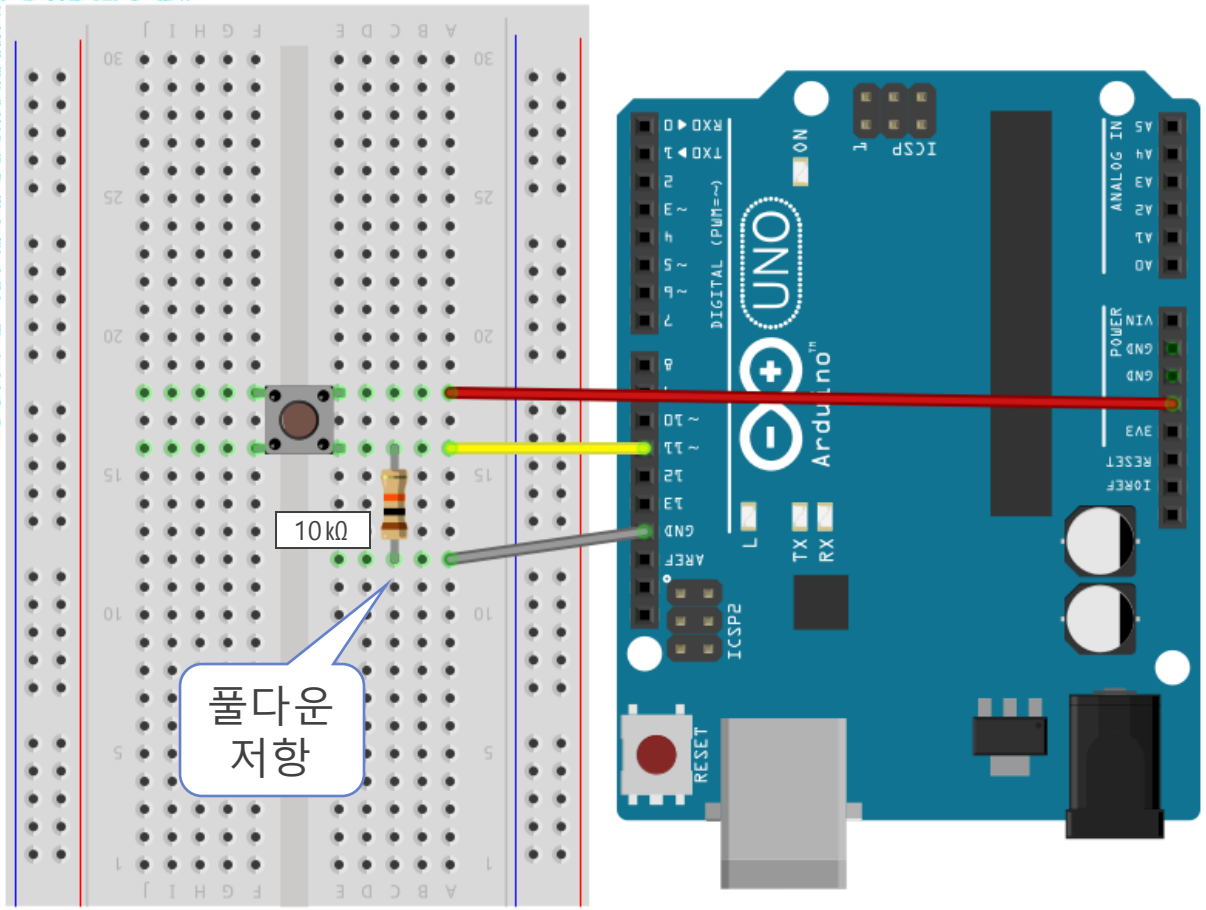


- 회로도

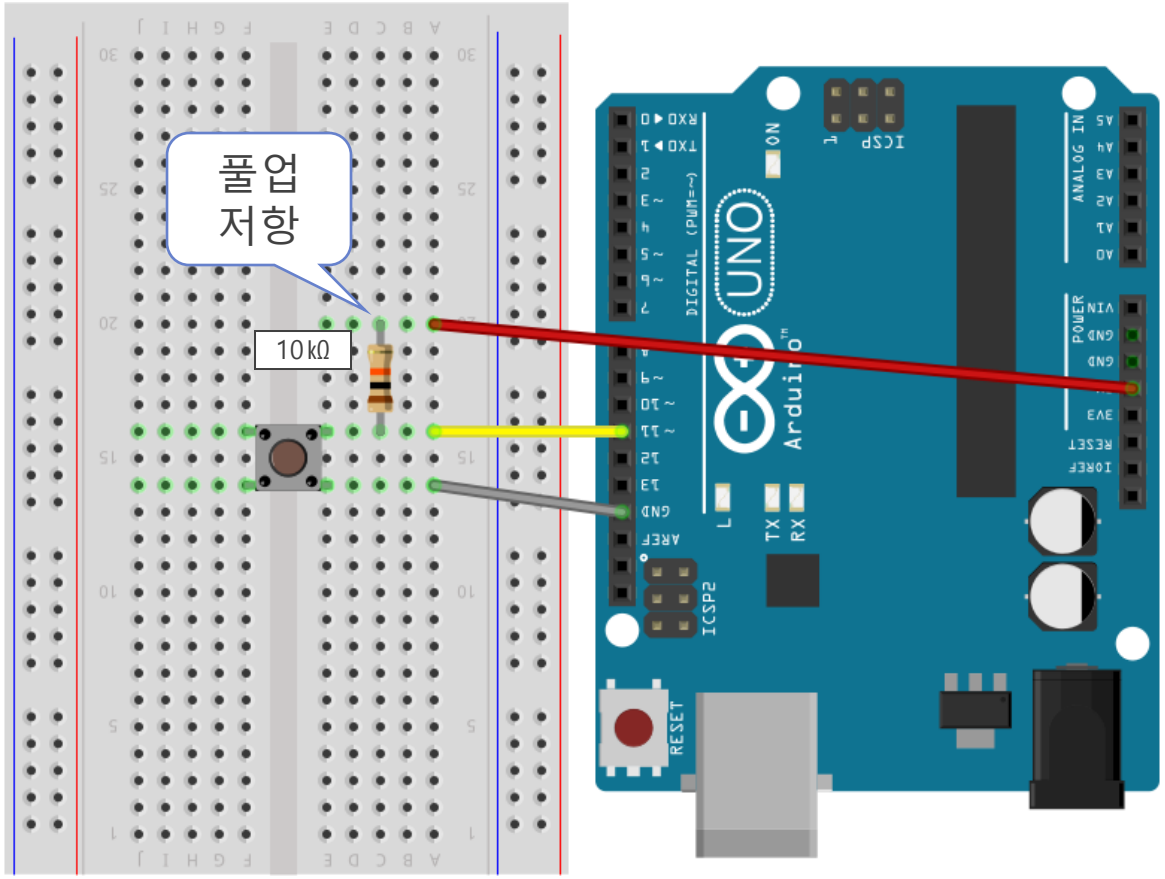


버튼 설계 정리

- 누르면 1(5V)되는 버튼



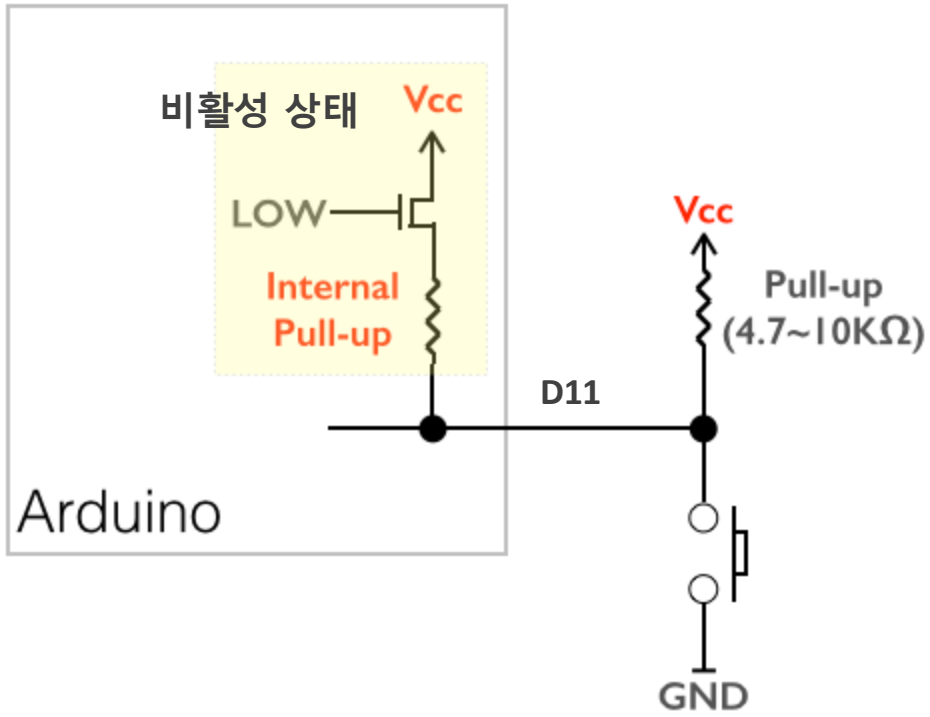
- 누르면 0(0V)되는 버튼



아하!
버튼을
사용하려면
저항을 붙여야
하는구나! ???

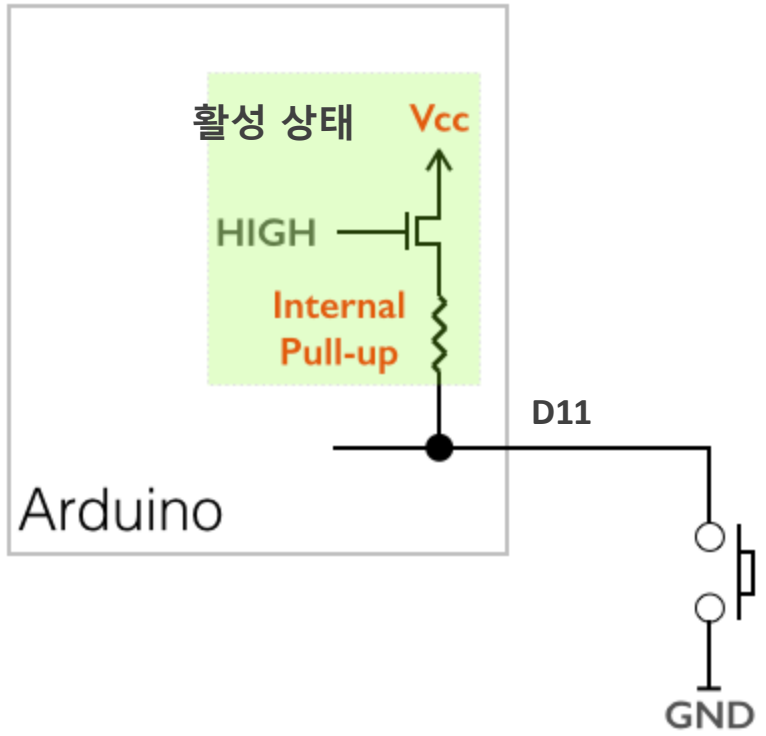
INPUT_PULLUP

- pinMode(11, INPUT);



- ACTIVE LOW(눌리면 0임)

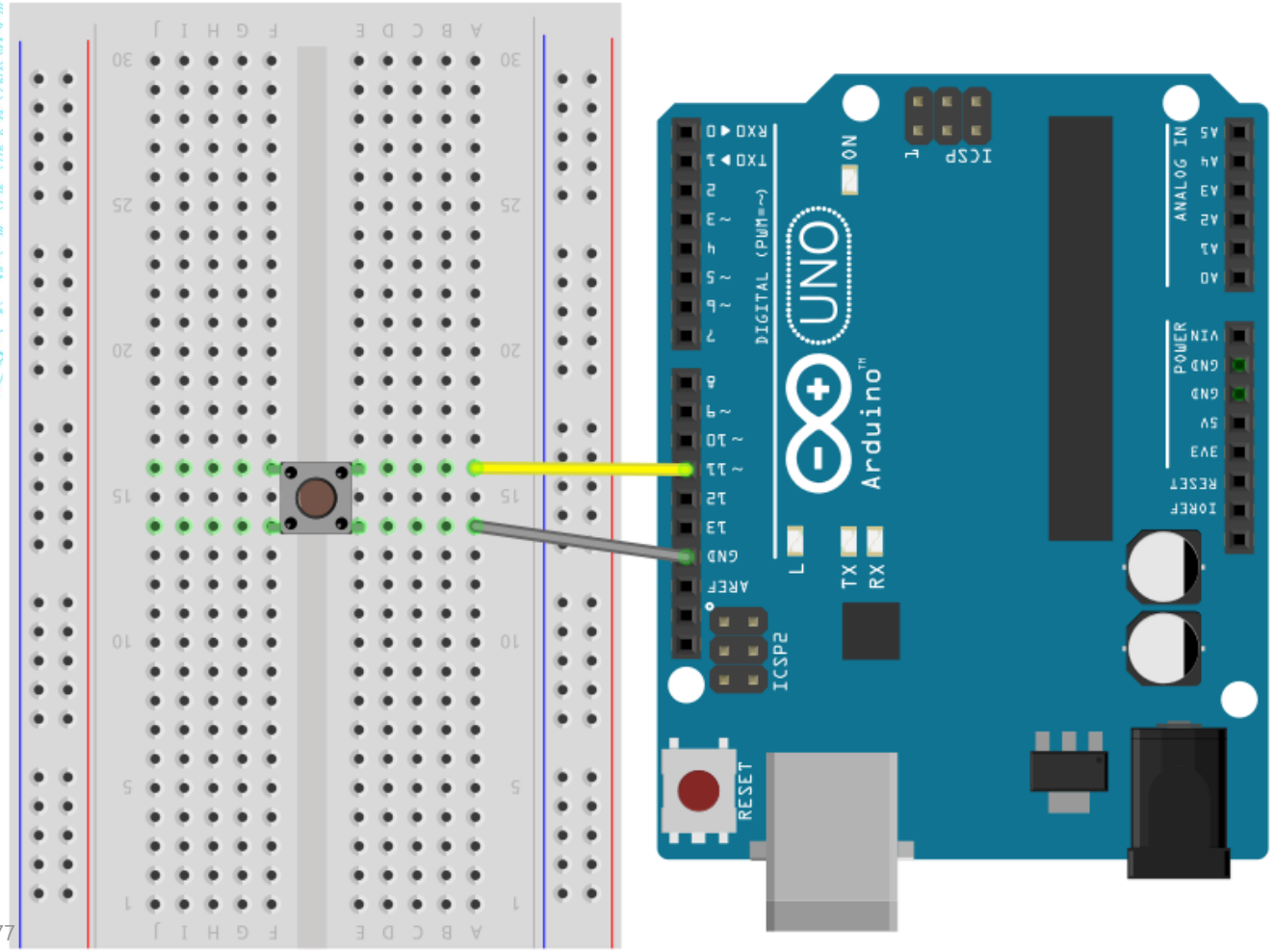
- pinMode(11, INPUT_PULLUP);



- 내부 회로를 사용하므로 Vcc 연결 없이 D11-Switch-GND로 연결하면 ACTIVE LOW인 버튼 구성 끝

INPUT_PULLUP 버튼

- 내부 풀업 저항 사용



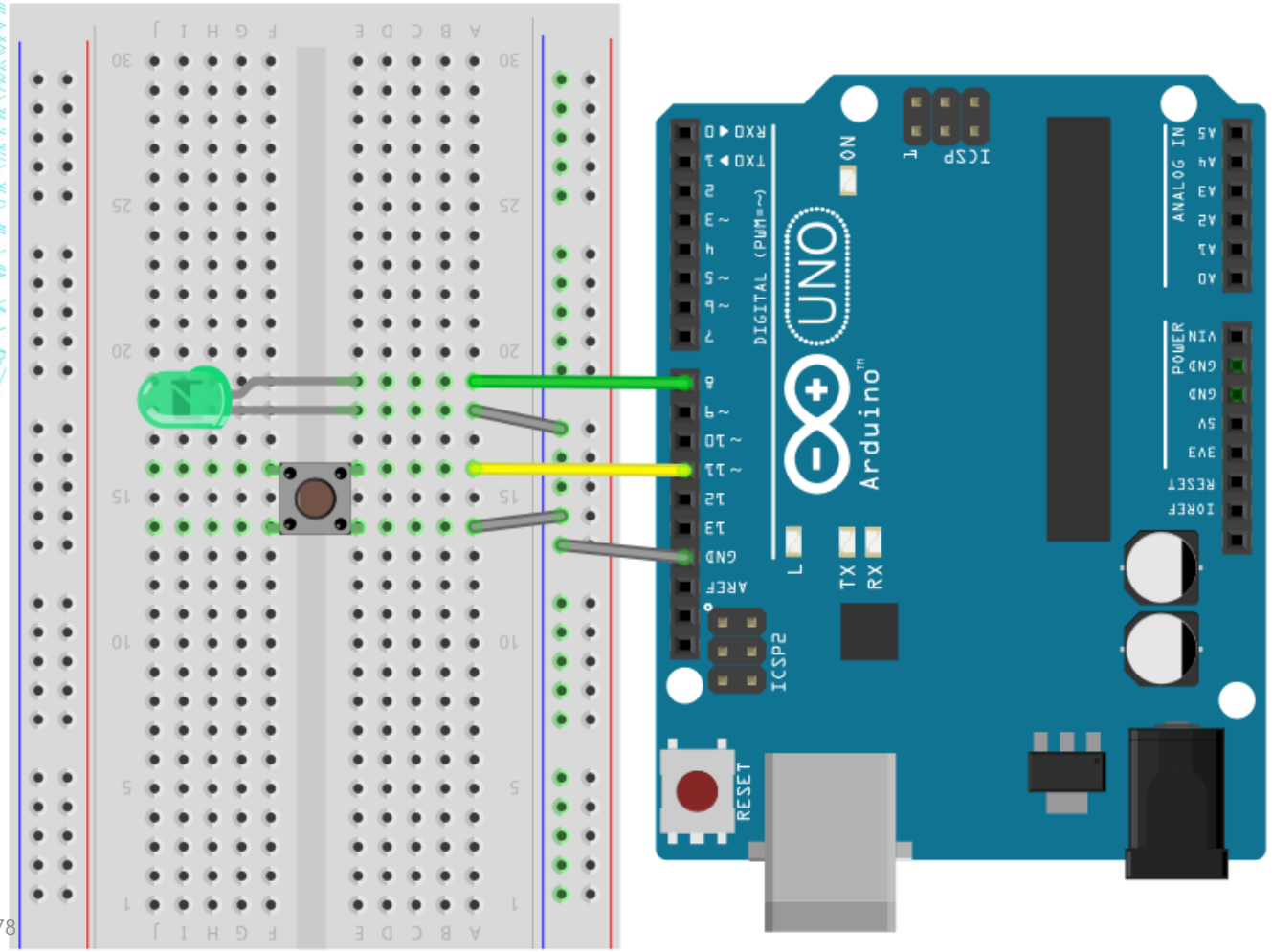
- 소스코드

2-1-1.ino

```
#define BTN 11

void setup() {
  pinMode(BTN, INPUT_PULLUP);
  Serial.begin(9600);
}

void loop() {
  bool r = digitalRead(BTN);
  Serial.println(r);
  delay(100);
}
```

INPUT_PULLUP 버튼

- 내부 풀업 저항 사용



- 소스코드

2-2-1.ino

```
//누르면 LED 켜지도록…
#define BTN 11
#define LED 8

void setup() {
  pinMode(BTN, INPUT_PULLUP);
  pinMode(LED, OUTPUT);
}

void loop() {
  bool btn = digitalRead(BTN);
  digitalWrite(LED, !btn);
}
```

토글 버튼의 구현

- 실습퀴즈

  아래와 같이 작동하는 버튼을 구현하시오.

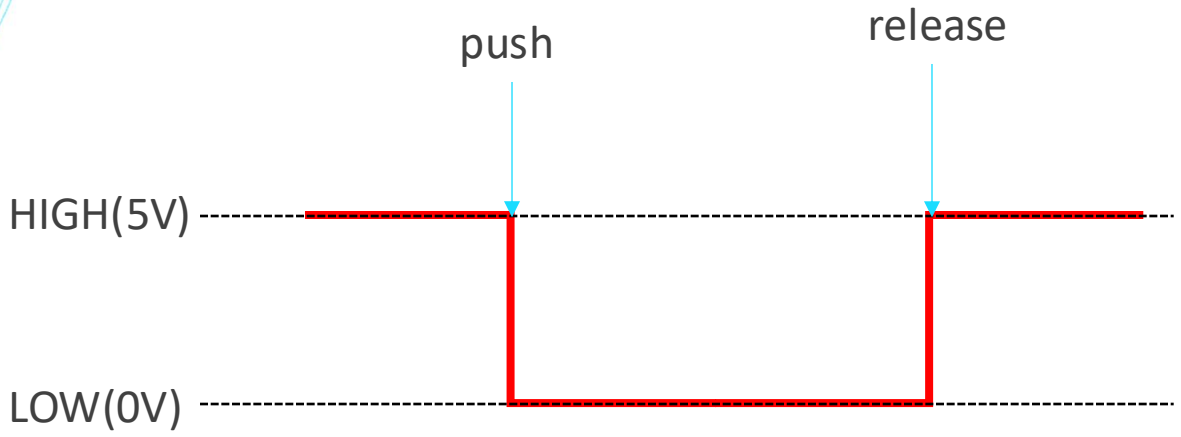
  - 버튼을 한 번씩 누를 때 마다
  - 8번 LED의 켜짐과 꺼짐이 반전

- 소스코드

```
#define LED  8
#define BTN 11

void setup() {



}

void loop() {




}
```

토글 버튼의 구현

- INPUT_PULLUP 버튼의 상태도



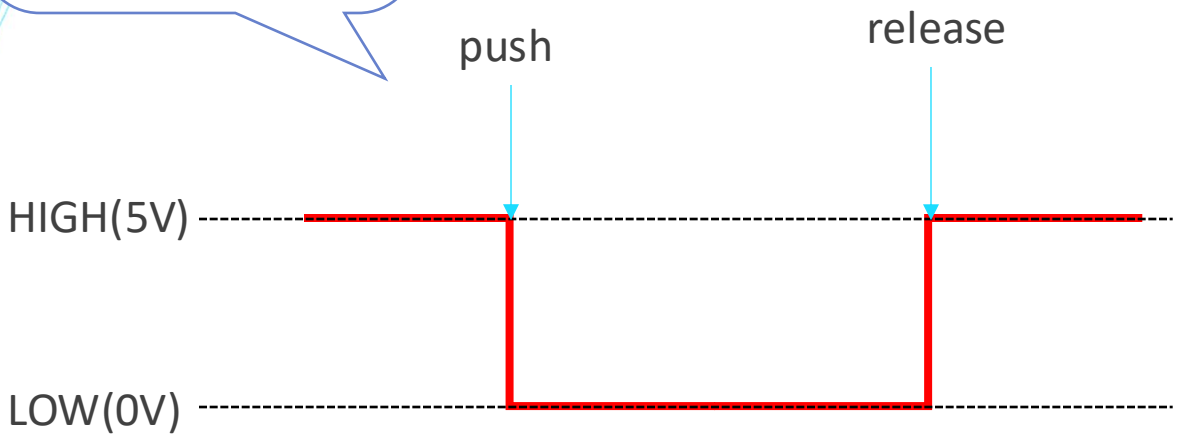
toggle1.ino

```
#define LED  8
#define BTN 11

void setup() {
  pinMode(LED, OUTPUT);
  pinMode(BTN, INPUT_PULLUP);
}

void loop() {
  static bool btn_pre = HIGH;
  bool btn_now = digitalRead(BTN);

  if(btn_pre == HIGH && btn_now == LOW) {
    digitalWrite(LED, !digitalRead(LED));
  }
  btn_pre = btn_now;
}
```

토글 버튼의 구현

- INPUT_PULLUP 버튼의 상태도

이론적으로는 완벽하지만 실제 작동은 완벽하지 않음. 그 이유를 확인해 보자.

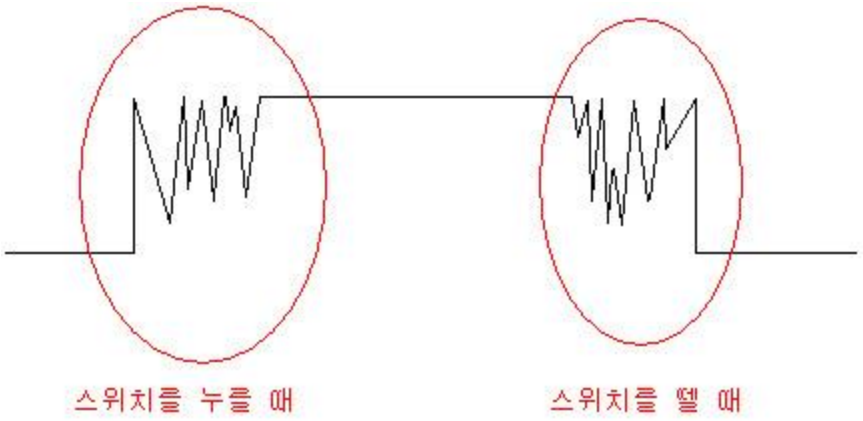


toggle2.ino

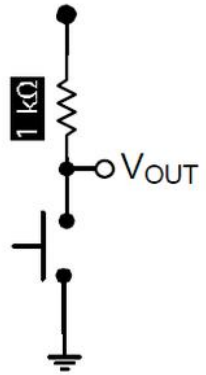
```
#define LED  8
#define BTN 12

void setup() {
  Serial.begin(9600);
  pinMode(LED, OUTPUT);
  pinMode(BTN, INPUT_PULLUP);
}

void loop() {
  static int n=0;
  static bool btn_pre = true;
  bool btn_now = digitalRead(BTN);

  if(btn_pre == HIGH && btn_now == LOW) {
    Serial.println(n++);
    digitalWrite(LED, !digitalRead(LED));
  }
  btn_pre = btn_now;
}
```

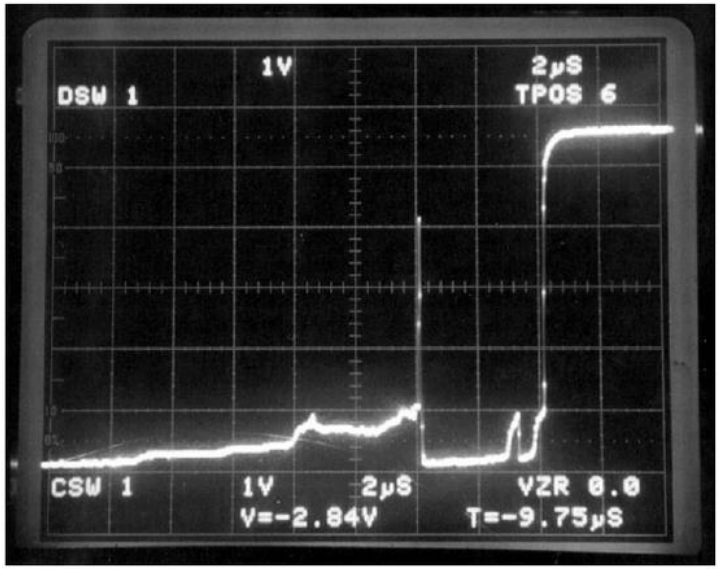
채터링(chattering)

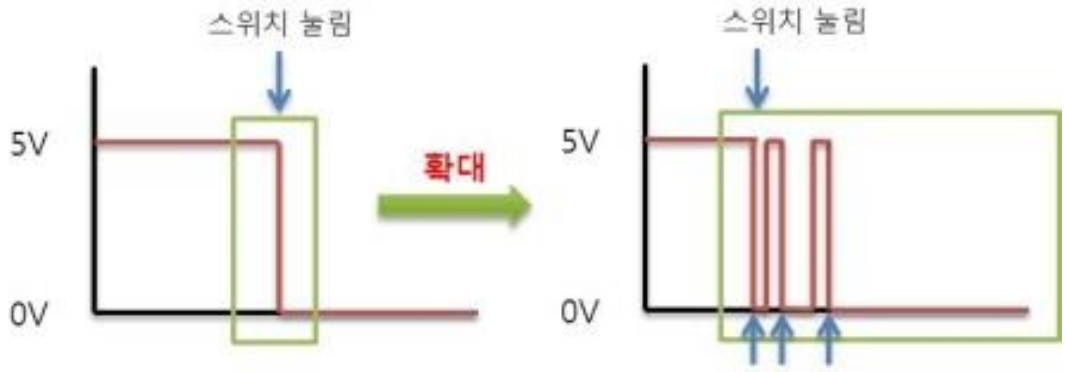
- Switch bounce 현상



- **Switch Bounce**



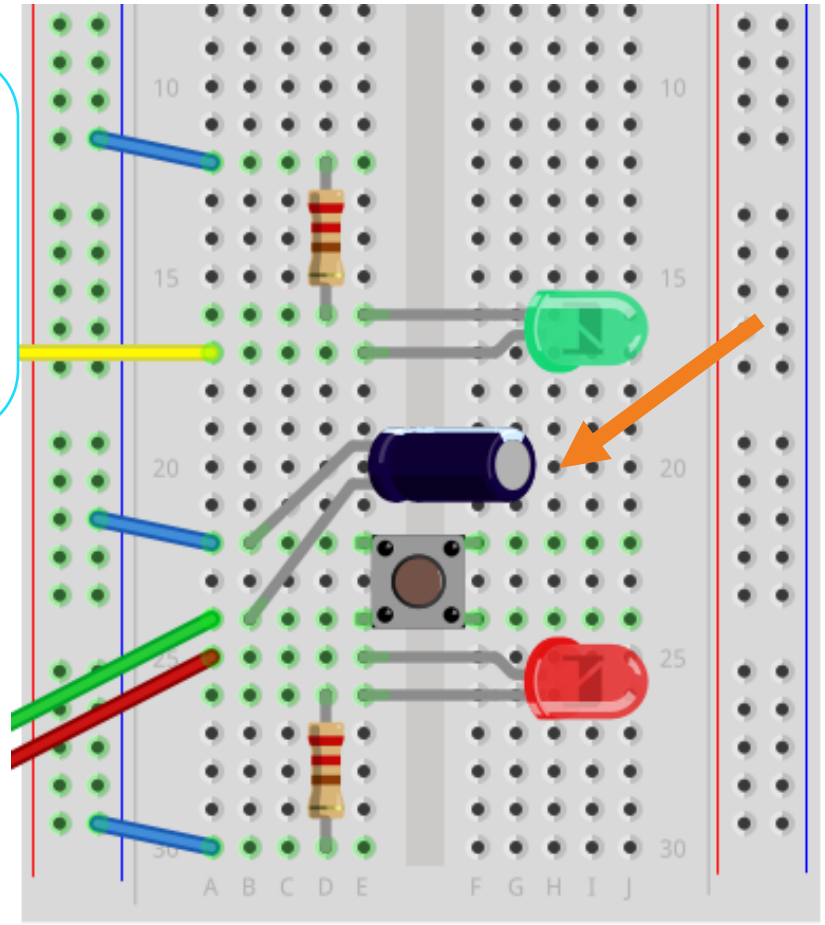




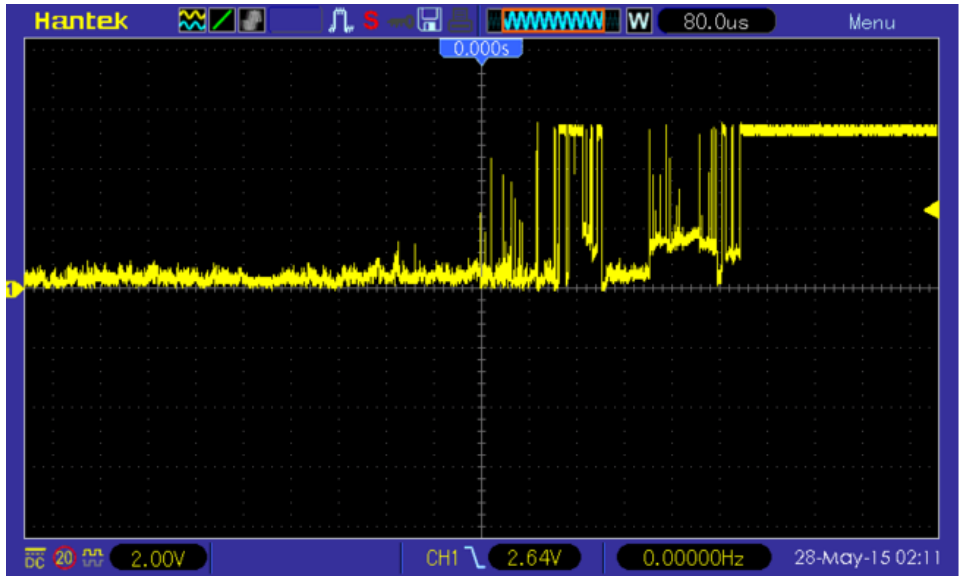
채터링 보정

- 하드웨어적인 해결책
  - 10uF 콘덴서 추가

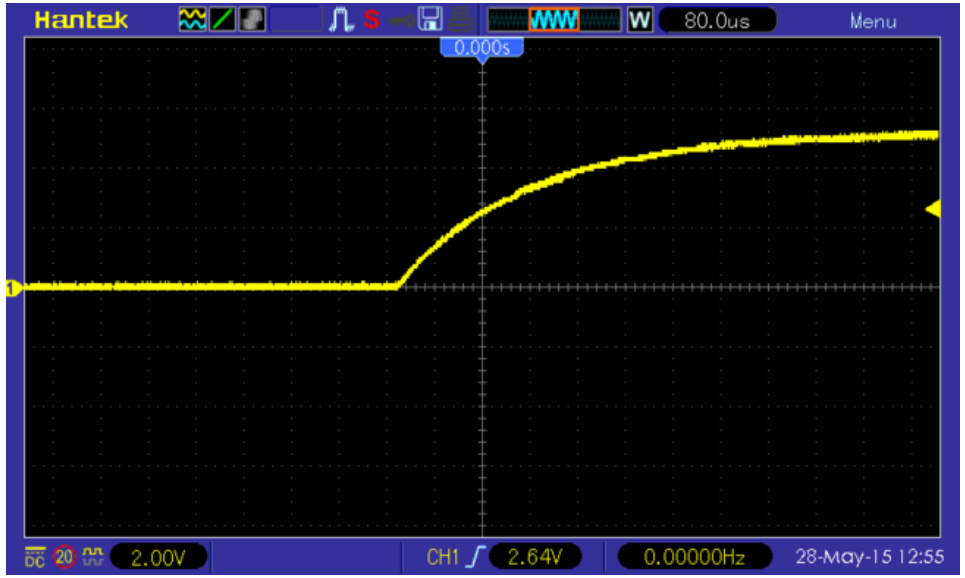
채터링 문제는
해결되었지만
버튼의 반응속도가
느려지는 부작용이
발생한다.



- 사용 전



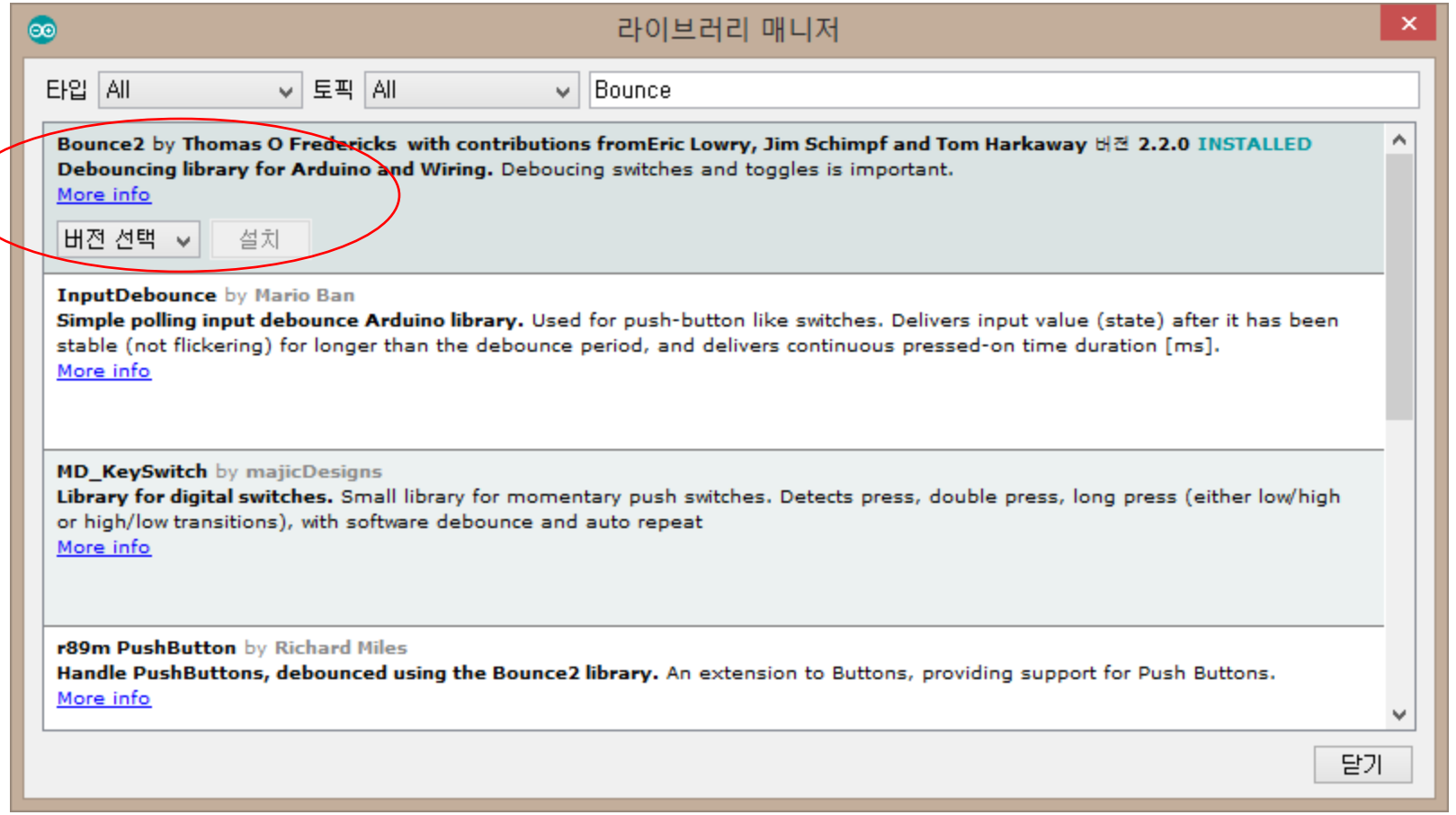
- 사용 후



채터링 현상은 사라졌지만 Rise time이 늘어나는 부작용 발생

채터링 보정(소프트웨어적인 방법)

- S/W 적인 해결책
  - Bounce2 library



Bounce2 library

- https://github.com/thomasfredericks/Bounce2/wiki
- **Method**
  - **void interval(unsigned long interval)**
    - Sets the debounce time in milliseconds.
  - **void attach(int pin)**
    - Sets the pin and matches the internal state to that of the pin. **Only attach the pin once you have done setting the pin up** (for example setting an internal pull-up).

pinMode 설정 먼저 하고 attach 호출

  - **bool update()**
    - Because **Bounce does not use interrupts, you have to "update" the object before reading its value** and it has to be done as often as possible (that means to include it in your loop()). **The update() method updates the object and returns true (1) if the pin state changed. False (0) if not.** Only call update() once per loop().

Bounce2 library

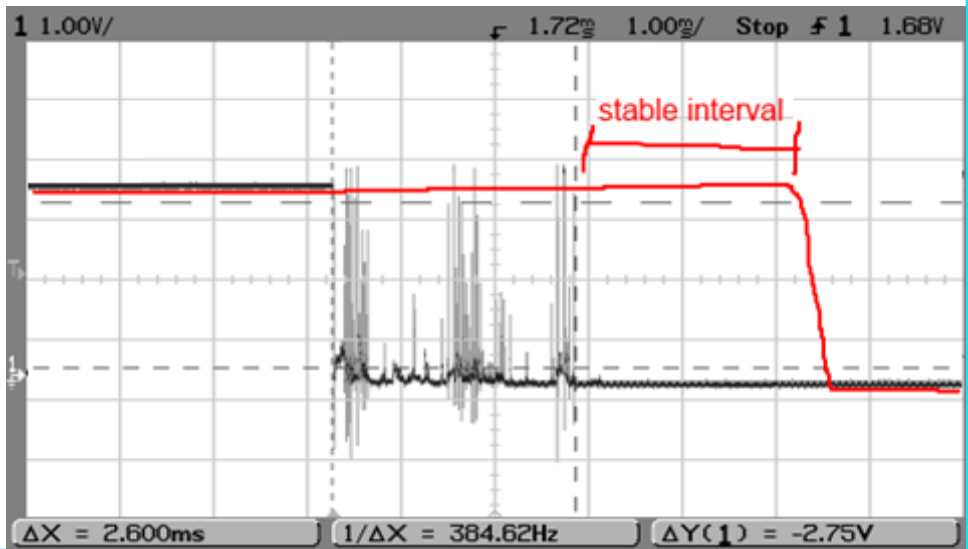
- **Method**
  - **bool read()**
    - Reads the updated pin state.
  - **bool fell()**
    - Returns true if pin signal transitions from high to low.
  - **bool rose()**
    - Returns true if signal transitions from low to high.
  - **bool risingEdge()**
    - Depreciated. An alias for rose(). For compatibility with Bounce 1.0.
  - **bool fallingEdge()**
    - Depreciated. An alias for fell(). For compatibility with Bounce 1.0.

채터링 보정(소프트웨어적인 방법)

2-6-1a.ino

```
#include <Bounce2.h>
#define BTN 11
#define LED 13

Bounce btn = Bounce();

void setup() {
  // BTN 핀을 INPUT_PULLUP으로 설정
  pinMode(BTN, INPUT_PULLUP);

  // BTN 핀을 debouncer에 부착
  btn.attach(BTN);
  btn.interval(50); // interval in ms

  //Setup the LED :
  pinMode(LED, OUTPUT);
}
```

```
void loop() {
  btn.update();           // 버튼 상태 업데이트
  bool res = btn.read();  // 버튼 값 읽기

  // INPUT_PULLUP 버튼은 안 눌렀을 때 HIGH 이다.
  if(res==HIGH)
    digitalWrite(LED, LOW);
  else
    digitalWrite(LED, HIGH);
}
```

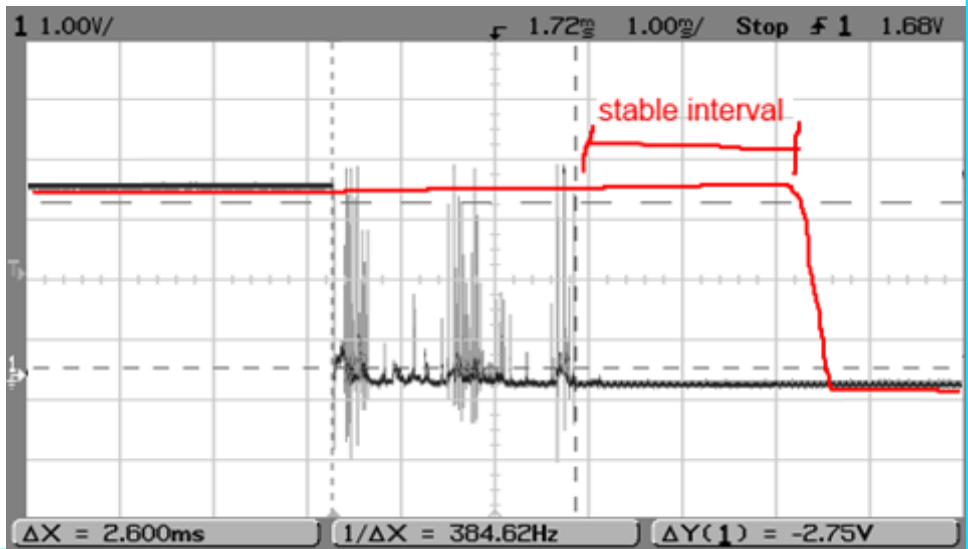


채터링 보정(소프트웨어적인 방법)

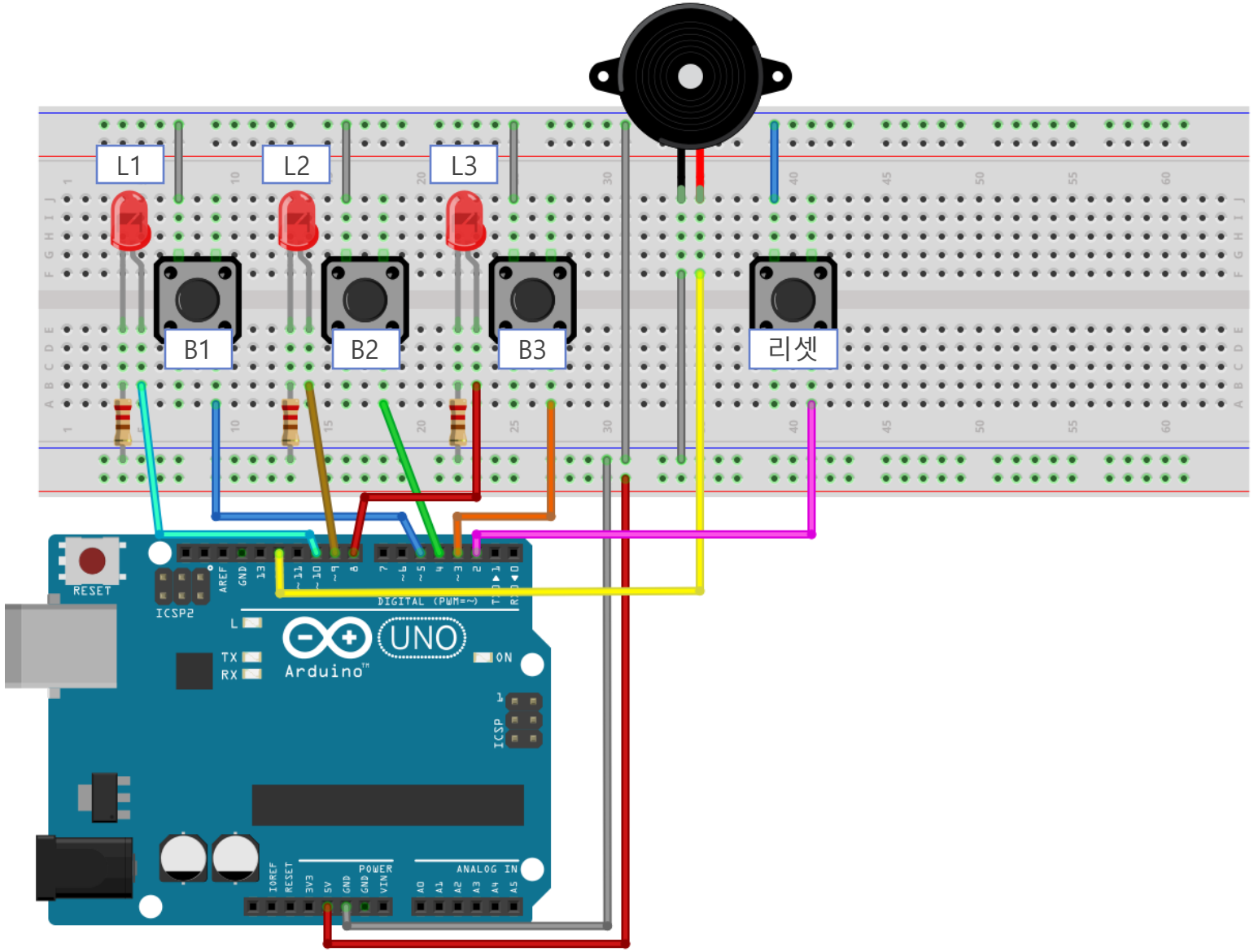
2-6-3a.ino

```
#include <Bounce2.h>
#define BTN 11
#define LED 13

Bounce btn = Bounce();

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

  // BTN 핀을 INPUT_PULLUP으로 설정
  pinMode(BTN, INPUT_PULLUP);

  // BTN 핀을 debouncer에 부착
  btn.attach(BTN);
  btn.interval(50);

  // Setup the LED :
  pinMode(LED, OUTPUT);
}
```

```
void loop() {
  static int n=0;
  btn.update();

  if( btn.fell() ) {  // transition from HIGH to LOW
    digitalWrite(LED, !digitalRead(LED));
  }

  if( btn.rose() ) {  // transition from LOW to HIGH
    Serial.println(n++);
  }
}
```



장학퀴즈 경쟁버튼 만들기



장학퀴즈 경쟁버튼 만들기

- 기기 동작
  - 초기상태 : 모든 LED는 꺼진 상태이고, 부저에서는 아무 소리도 나지 않으며, 각 버튼은 입력을 기다리고 있으며 어떤 버튼이 가장 빨리 눌린 버튼인지 알아내기 위해 대기하고 있다.
  - 각 버튼은 저마다의 톤을 가지고 있다. 첫 번째 버튼은 4옥타브의 '도', 두 번째 버튼은 '레', 세 번째 버튼을 누르면 '미' 소리가 나온다.
  - 가장 먼저 눌린 버튼의 왼쪽 옆에 있는 LED에 불이 들어와야 한다. 어느 버튼이 먼저 눌린 것인지 알 수 없을 정도로 거의 동시에 눌린 경우에는 두 개 이상의 버튼이 먼저 눌린 버튼으로 인정 될 수 있다. (즉 두 개 이상의 LED가 켜질 수 있다)
  - 승부가 결정되어 LED의 출력이 시작된 경우에는 버튼의 입력은 무시된다.
  - 리셋 버튼을 누르면 초기 상태로 되돌아간다.

장학퀴즈 경쟁버튼 만들기

- 실제 세품으로 만든다면 선택 가능한 버튼

[당일발송]
60mm 빨강색 원형 LED 아케이드 스위치 버튼 (돔 모양)
3,300원

[당일발송]
6칸형 전선연결 버튼형 커넥터
1,540원

[당일발송]
32mm 빨강색 반투명 사각 LED 아케이드 스위치 버튼
2,200원

[당일발송]
32mm 초록색 반투명 사각 LED 아케이드 스위치 버튼
2,200원

[당일발송]
46mm 노랑색 원형 LED 아케이드 스위치 버튼
2,200원

[당일발송]
46mm 녹색 원형 LED 아케이드 스위치 버튼
2,200원

[당일발송]
46mm 화이트 원형 LED 아케이드 스위치 버튼
2,200원

[당일발송]
39mm 화이트 삼각 LED 아케이드 스위치 버튼
1,540원

[당일발송]
100mm 녹색 원형 LED 아케이드 스위치 버튼 (돔 모양)
5,500원

[당일발송]
100mm 빨강색 원형 LED 아케이드 스위치 버튼 (돔 모양)
5,500원

[당일발송]
100mm 화이트 원형 LED 아케이드 스위치 버튼 (돔 모양)
5,500원

[당일발송]
한글보드: 디지털 푸쉬버튼 스위치 (노랑) / 택트 푸시버튼 / Button Module
1,650원
[사용예제]

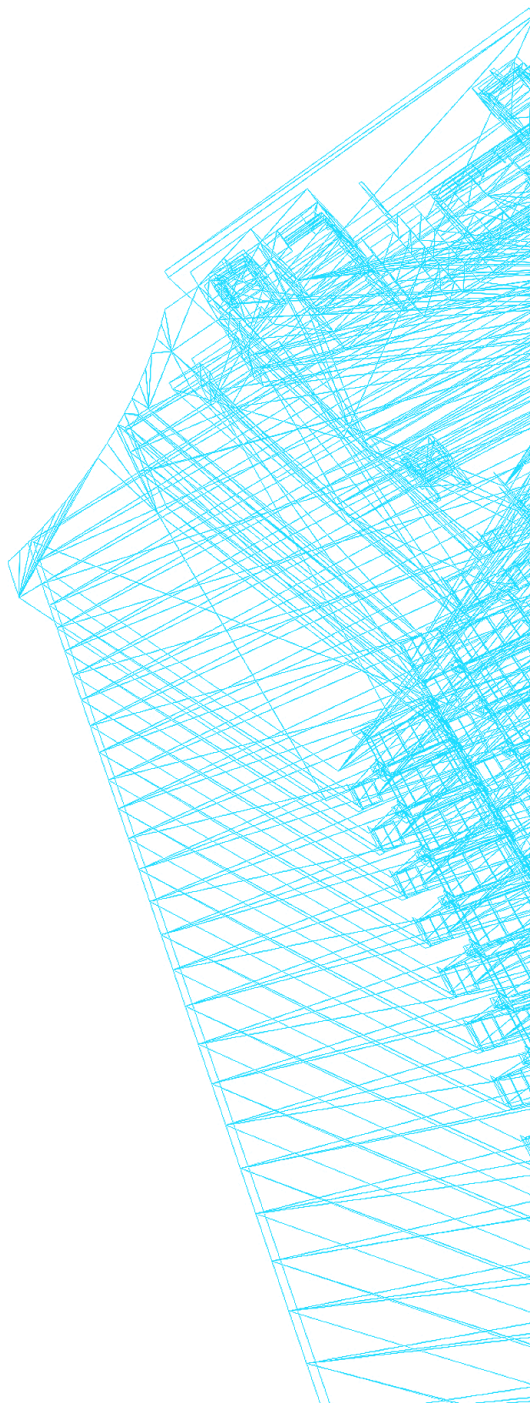
competition.ino

```
#define RESET   2
#define BTN1    3
#define BTN2    4
#define BTN3    5
#define LED1    8
#define LED2    9
#define LED3   10
#define BUZZER 12

bool isLocked = false; // 중복 눌림의 예방하기 위함
bool aryButton[] = { false, false, false };  // 버튼 눌림 정보
int  note[] = { 523, 587, 659 };  // 도레미
int i;

void setup() {
  for(i=LED1; i<=LED3; i++)
    pinMode(i, OUTPUT);

  for(i=RESET; i<=BTN3; i++)
    pinMode(i, INPUT_PULLUP);
}

void reset() {
  for(i=0; i<3; i++) {
    aryButton[i] = false;
    digitalWrite(LED1+i, LOW);  // LED off
  }
  isLocked  = false;
}
```

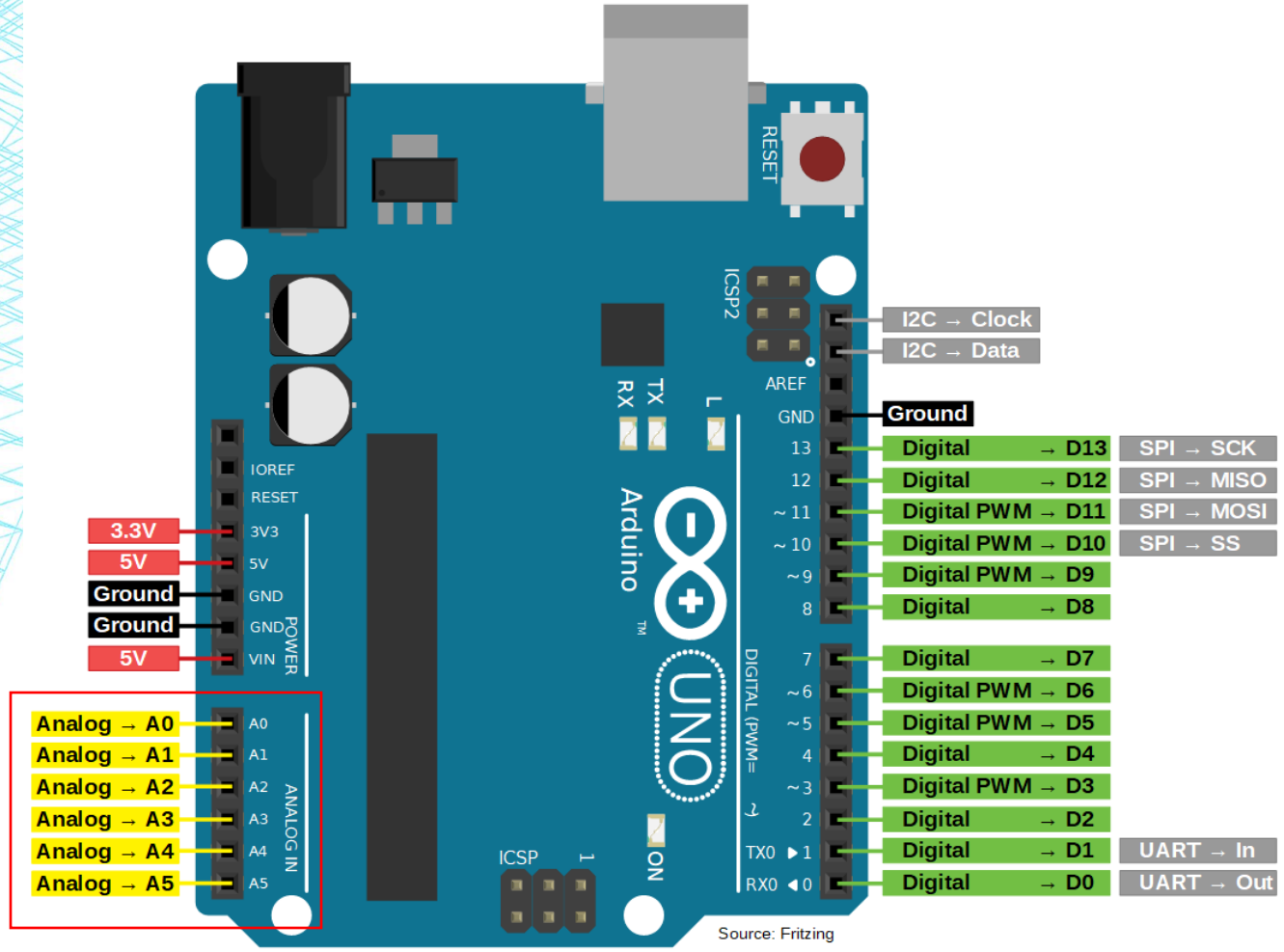
```
void button_monitor() {
  if(digitalRead(RESET) == LOW) {
    tone(BUZZER, 880, 300);
    reset();
  }

  if(isLocked)
    return;

  for(i=BTN1; i<=BTN3; i++) {
    bool r = digitalRead(i);  // 눌렀을 때, LOW가 됨
    aryButton[i-BTN1] = !r;
  }

  for(i=0; i<3; i++)
    if(aryButton[i] == true) {
      digitalWrite(LED1+i, HIGH);
      tone(BUZZER, note[i], 300);
      isLocked = true;
      pressedTime = millis();
    }
}

void loop() {
  button_monitor();
}
```

Analog Input

- ADC
- 아날로그 입력 실험
- CdS(조도센서) 연결



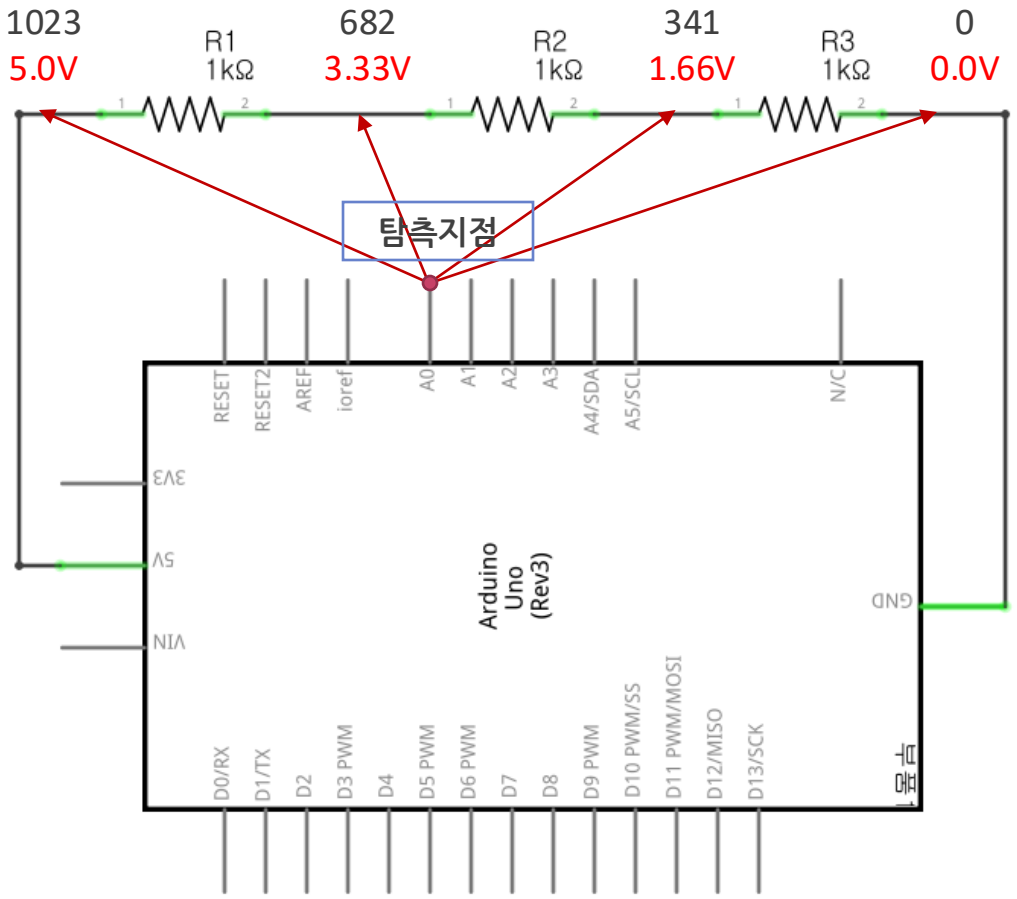
ADC(Analog to Digital Convertor)



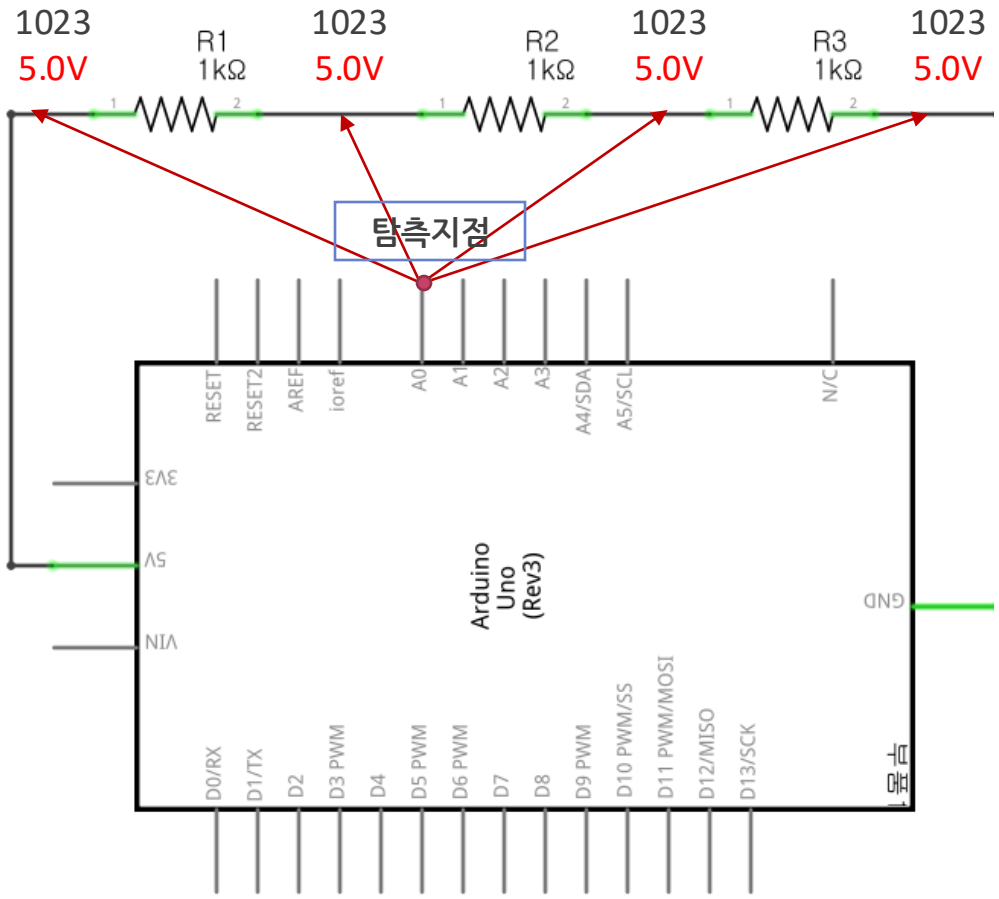
- ADC는 ANALOG IN 핀만 가능
- 핀에서 입력된 전압의 크기를 읽어냄
- analogRead(Ax) 함수 사용
- analogRead()는 pinMode()초기화 안해도 됨
- 10bit 분해능
- 0 ~ 5V 전압을 0 ~ 1023 숫자로 알려줌

아날로그 입력 실험

- 닫힌 회로

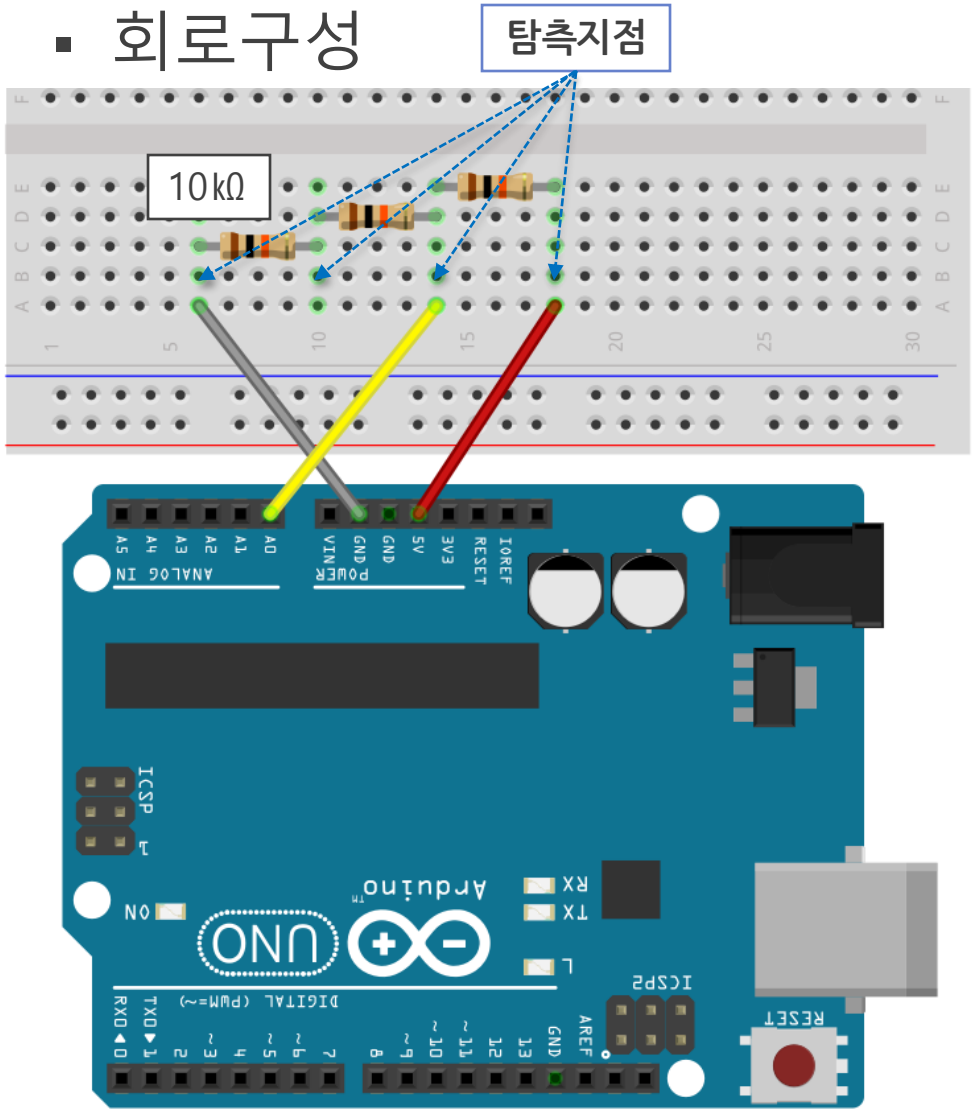


- 개방된 회로



아날로그 입력 실험

- 회로구성



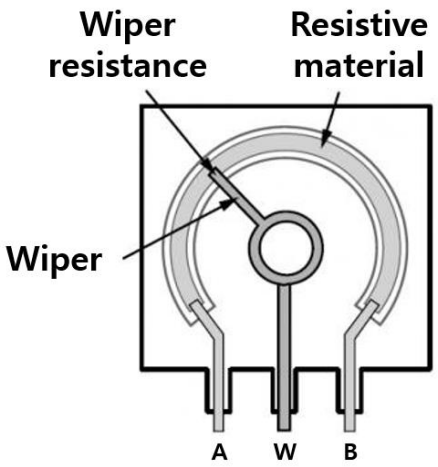
- 소스코드

3-1.ino

```
void setup() {
  Serial.begin(9600);
  Serial.println("a, V");
}

// A0에 연결된 선을 각 탐측지점에
// 꽂아 보면서 실험한다.
void loop() {
  int a = analogRead(A0);
  // 0~1023 범위의 값을 0~5범위로.
  double v = a/1023.0 * 5;

  Serial.print(a);
  Serial.print(", ");
  Serial.print(v);
  Serial.println("V");
  delay(200);
}
```

가변저항(potentiometer) 연결하기

- 회로구성



https://en.wikipedia.org/wiki/Potentiometer

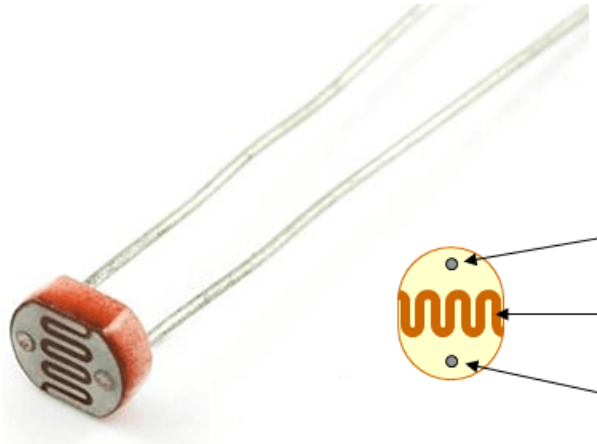
- 소스코드

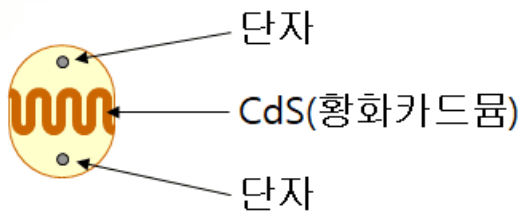
3-1.ino

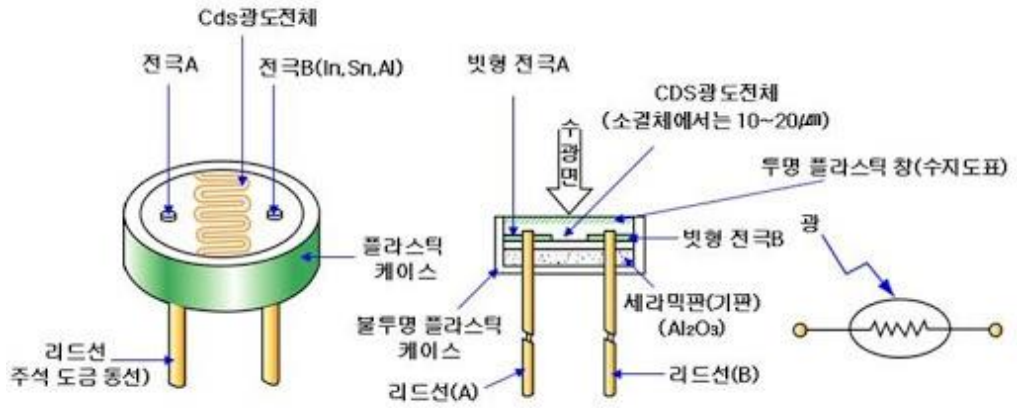
```
void setup() {
  Serial.begin(9600);
  Serial.println("a, V");
}

// A0에 연결된 선을 각 탐측지점에
// 꽂아 보면서 실험한다.
void loop() {
  int a = analogRead(A0);
  // 0~1023 범위의 값을 0~5범위로.
  double v = a/1023.0 * 5;

  Serial.print(a);
  Serial.print(", ");
  Serial.print(v);
  Serial.println("V");
  delay(200);
}
```

CdS(조도센서)

- CdS 외형과 특징



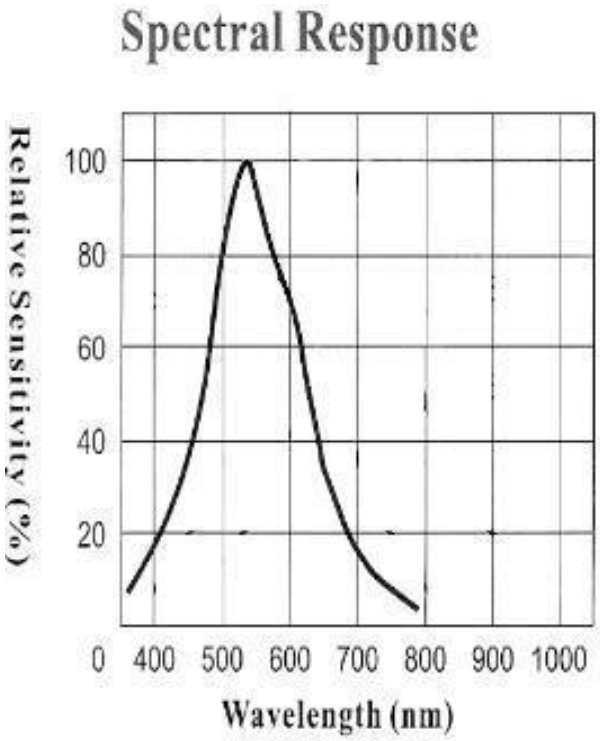


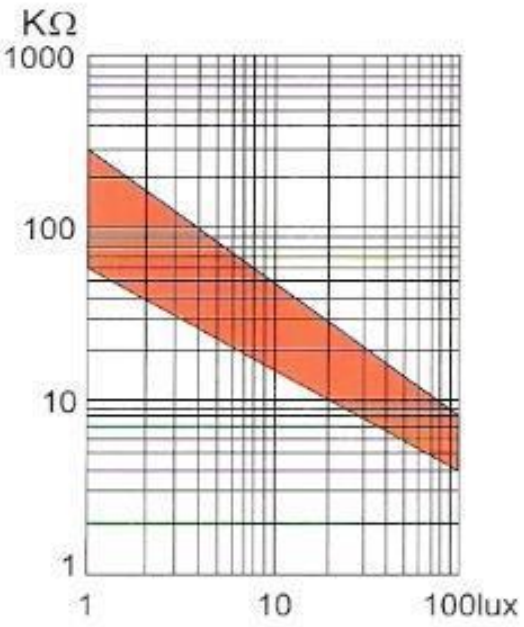


(a) CDS셀의 외관 모양 (b) CDS셀의 단면 구조 (플라스틱 케이스형) (c) 기 호

Specifications

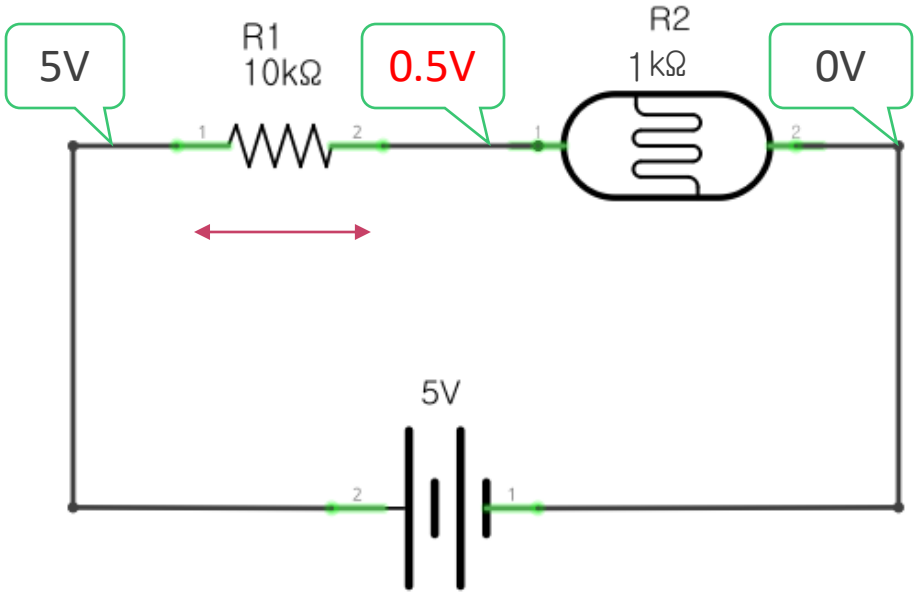
Model	Vmax (VDC)	Pmax (mW)	Ambient temp(℃)	Spectral peak (nm)	Light Resistance at 10Lux (KΩ)	Dark Resistance (MΩ)	Gamma value at 100-10Lux	Response Time (ms)	
								Rise Time	Decay time
GL5516	150	90	-30~+70	540	5-10	0.2	0.6	30	30
					1000Ω	1MΩ			





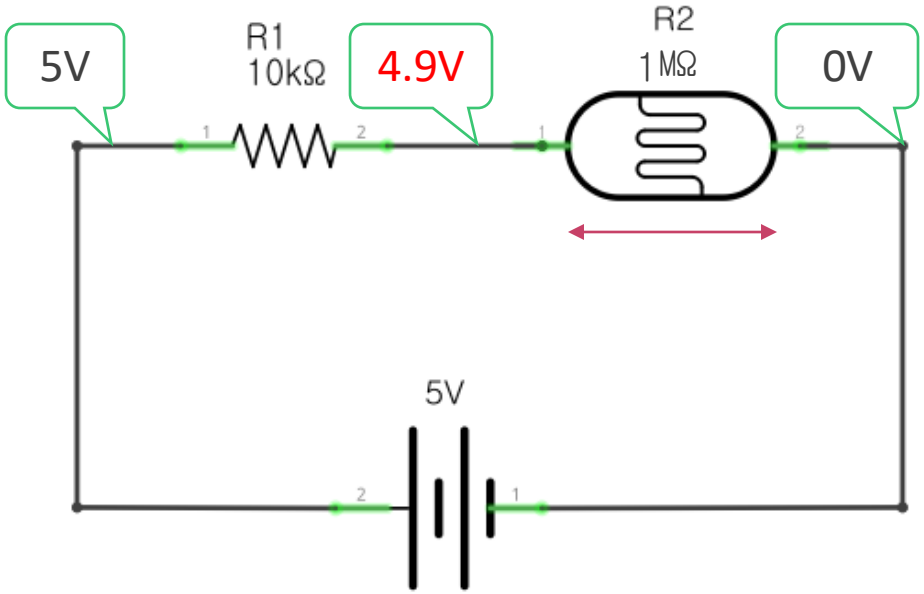
CdS(조도센서) 연결하기1 – 저항 먼저

- 밝은 곳



- 중간 지점의 전압이 낮게 나온다.

- 어두운 곳

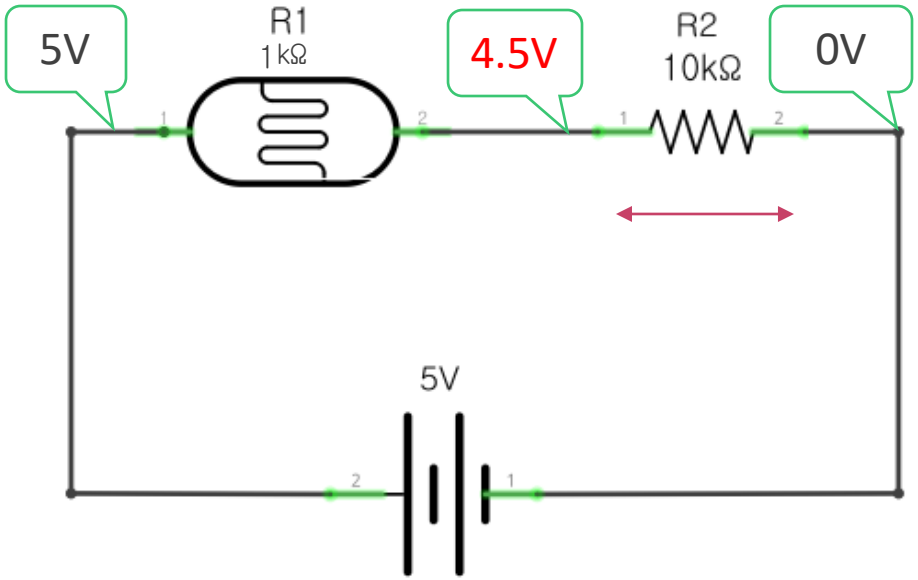


- 중간 지점의 전압이 높게 나온다.

※ 밝을 수록 작은 숫자, 어두울 수록 큰 숫자가 나오는 설계

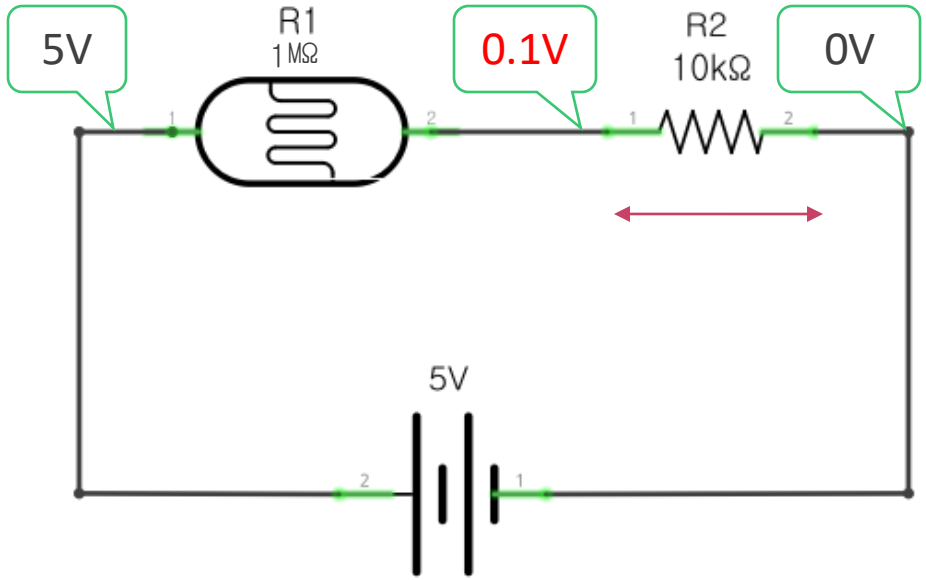
CdS(조도센서) 연결하기2 – 센서 먼저

- 밝은 곳



- 중간 지점의 전압이 높게 나온다.

- 어두운 곳

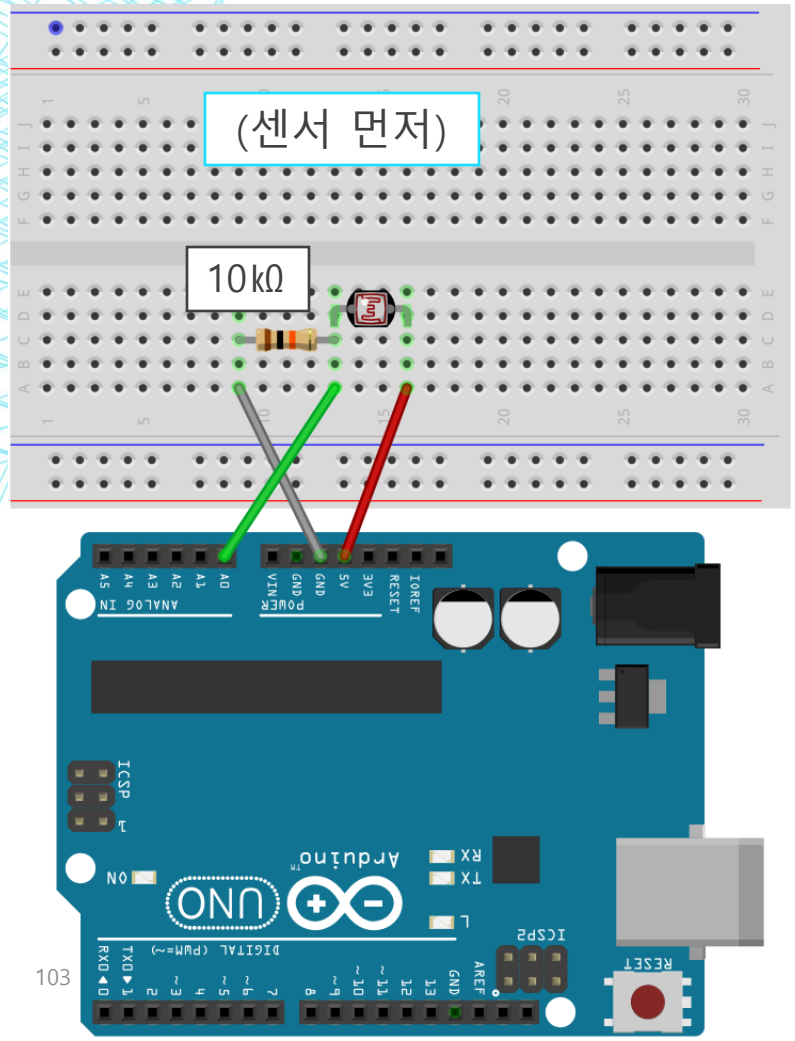


- 중간 지점의 전압이 낮게 나온다.

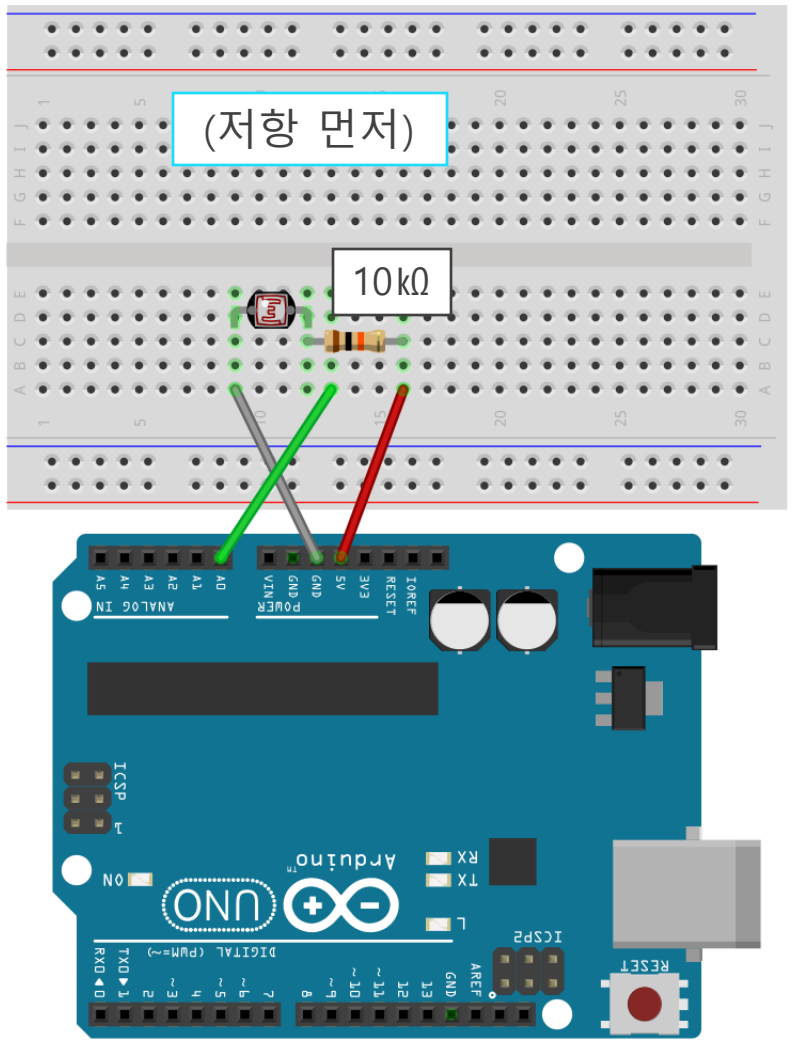
※ 밝을 수록 큰 숫자, 어두울 수록 작은 숫자가 나오는 설계

CdS(조도센서) 연결하기

- 밝을 수록 큰 숫자 회로



- 어두울 수록 큰 숫자 회로



- 소스코드

3-2.ino

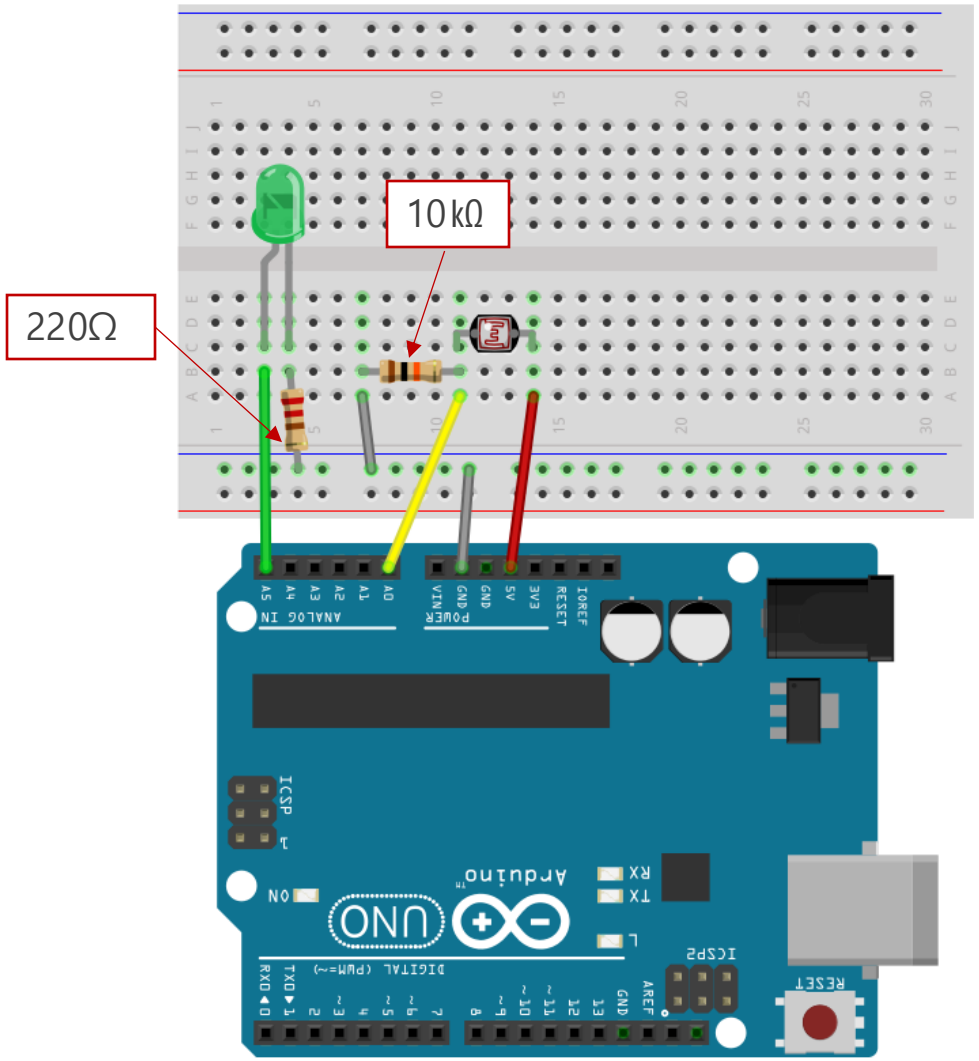
```
void setup() {
  Serial.begin(9600);
}

void loop() {
    int v = analogRead(A0);
    Serial.println(v);
    delay(500);
}
```

Analog Input - 실습과제

- 조도센서와 LED를 조합하여 어두워지면 저절로 켜지는 가로등을 구현 하시오.
- 가로등을 실제로 가져올 수 없으므로 LED를 가로등이라 가정한다.

Analog Input – 실습과제

- 회로구성



- 소스코드

3-3.ino

```
#define CDS A0
#define LED A5

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

void loop() {
    ?
}
```

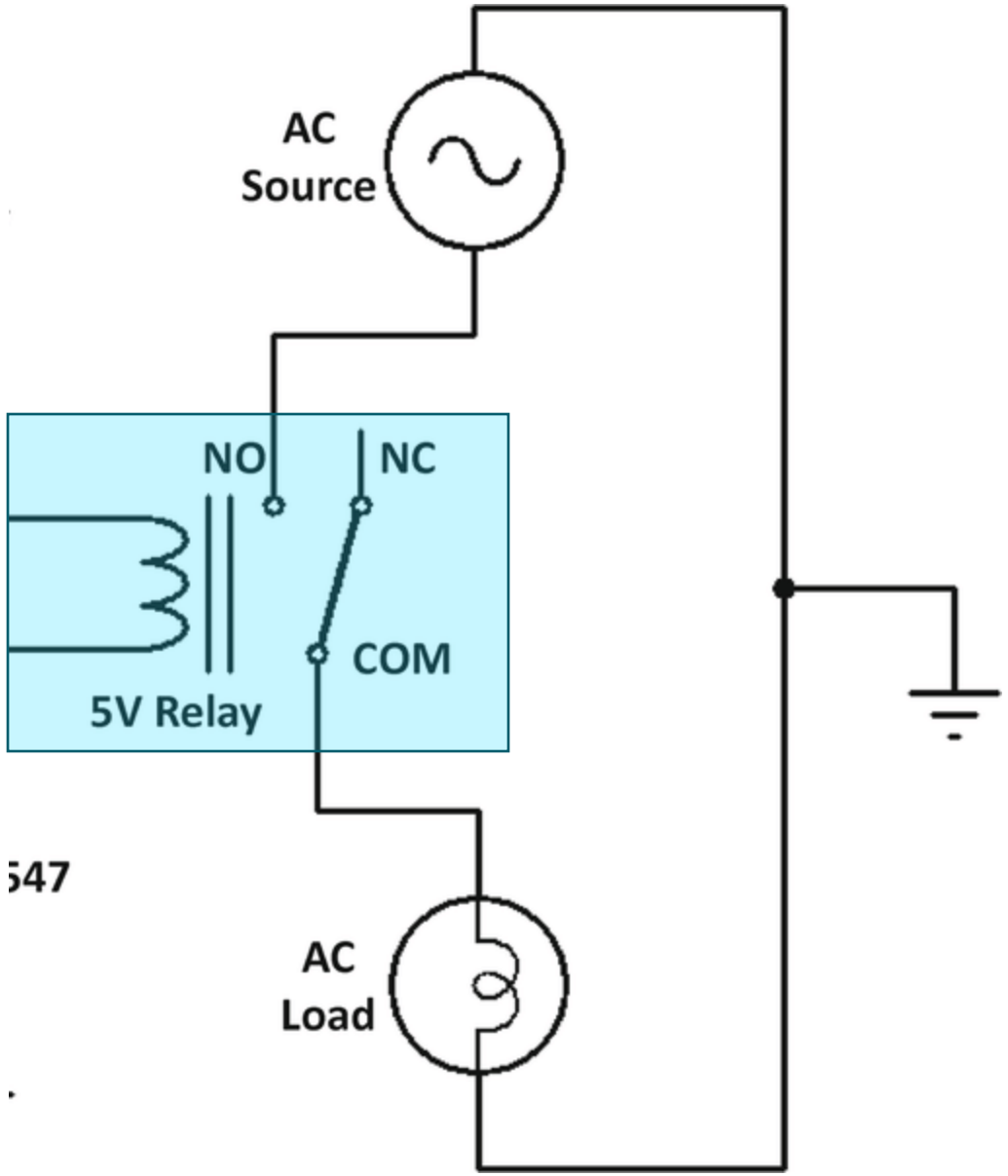
Relay

- 릴레이 소자
  - 작은 전압으로 큰 전압을 제어 가능
  - 아두이노에서 선풍기, 가정용 전등 등을 켜고 끌 수 있게 됨

구동전압
5VDC

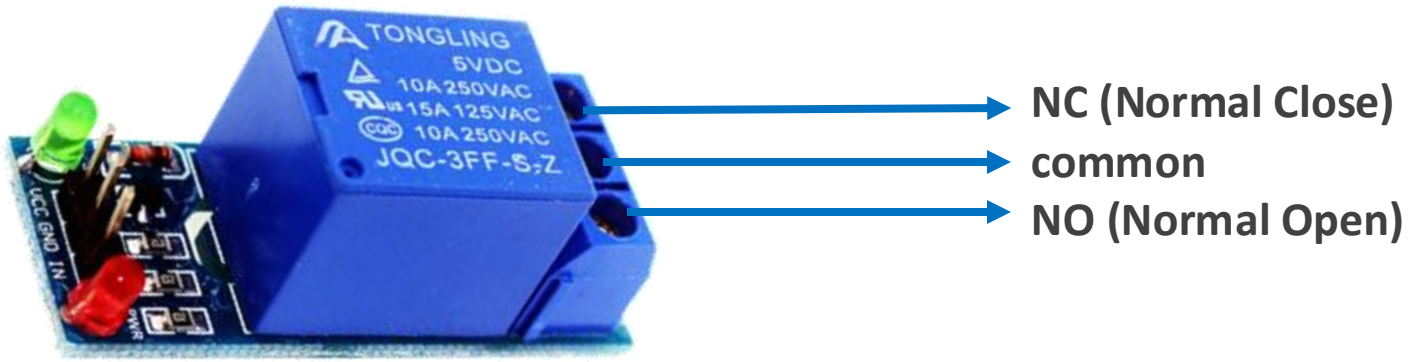
기본입력전압
10A 125VAC
10A 28VDC

최대입력전압
10A 250VAC
10A 30VDC

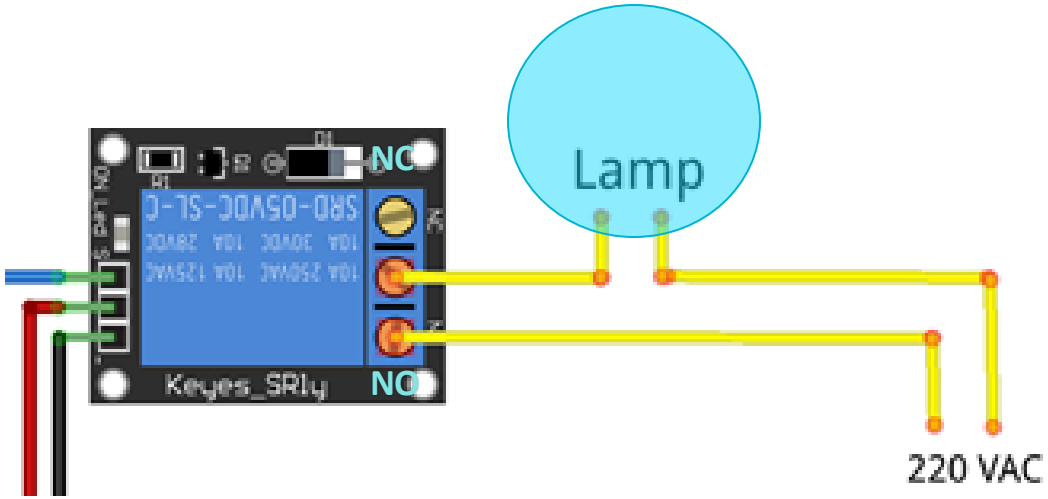


Relay

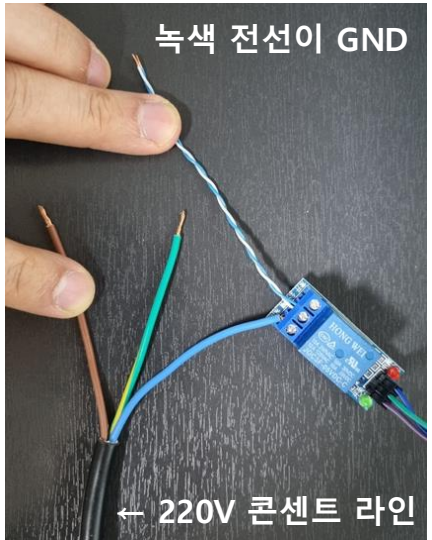
- 릴레이 모듈



- IN 입력에 대한 반응
  - 1이면 : common과 NO 단자가 연결
  - 0이면 : common과 NC 단자가 연결

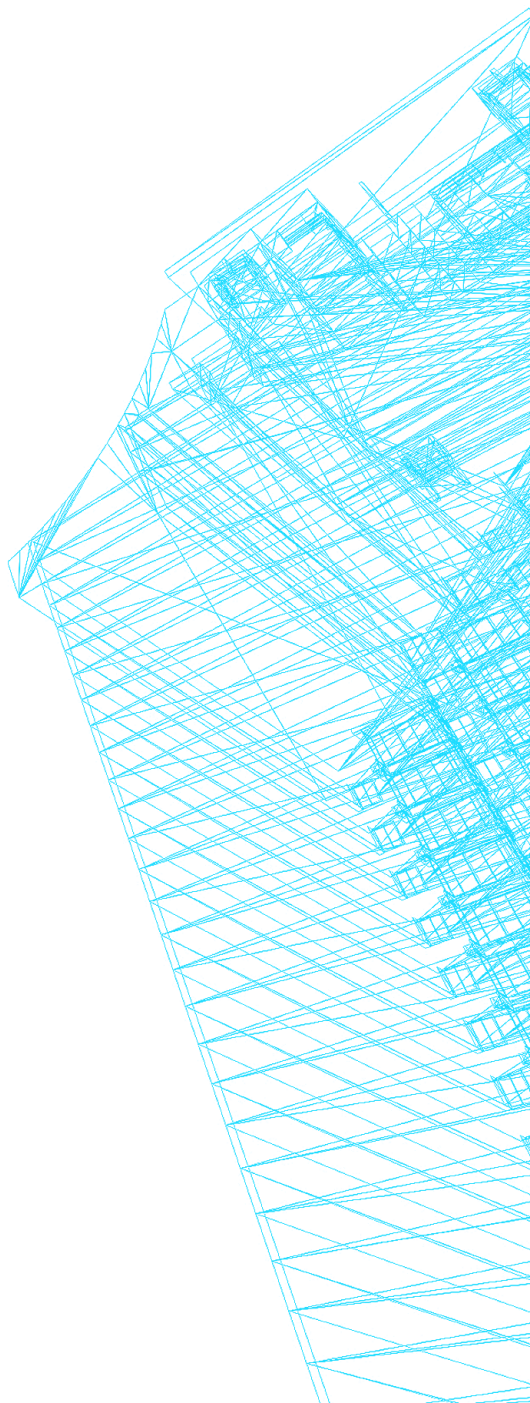


어두워지면 저절로 켜지는 가로등

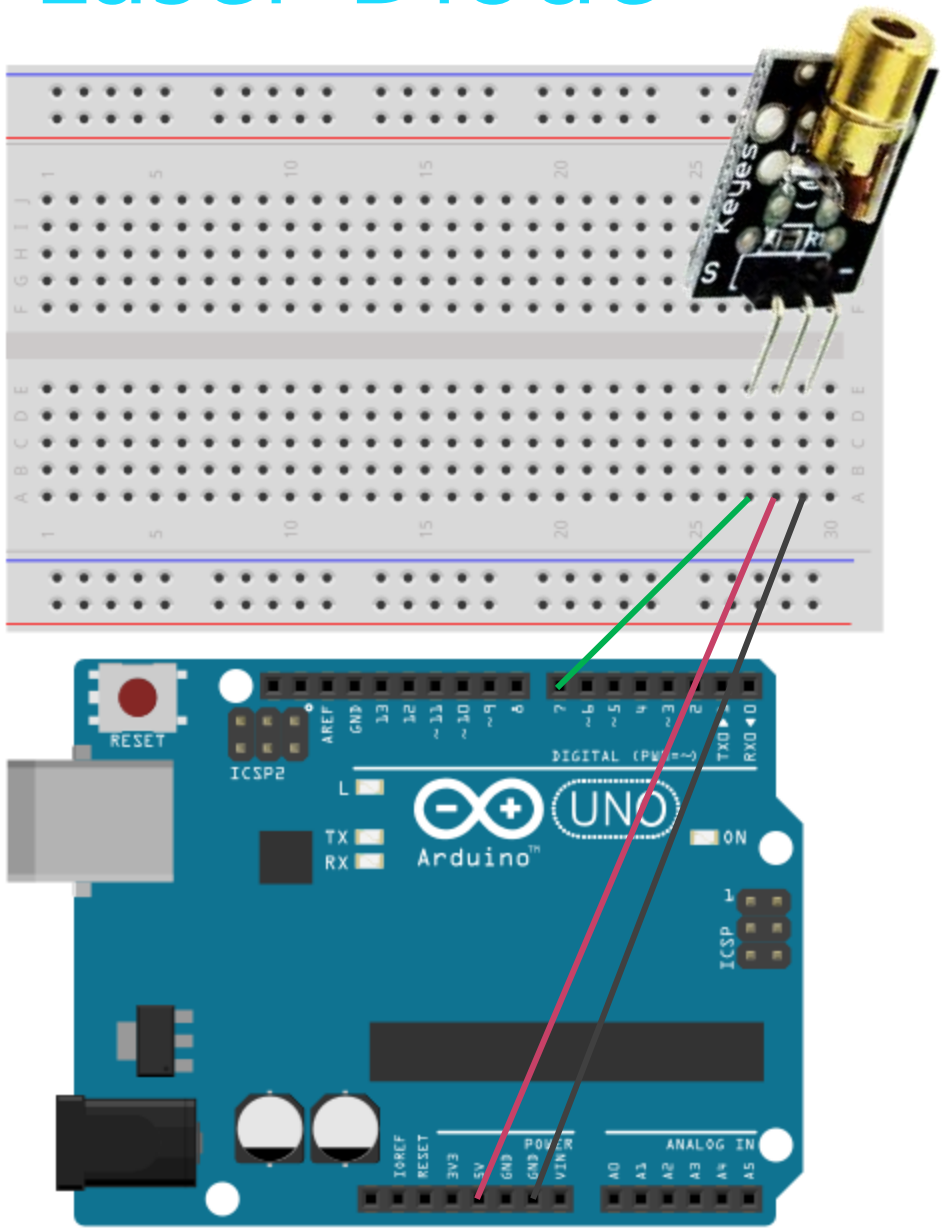


Digital Sensor Module

- Laser diode
- Tilt sensor
- PIR sensor
- Relay



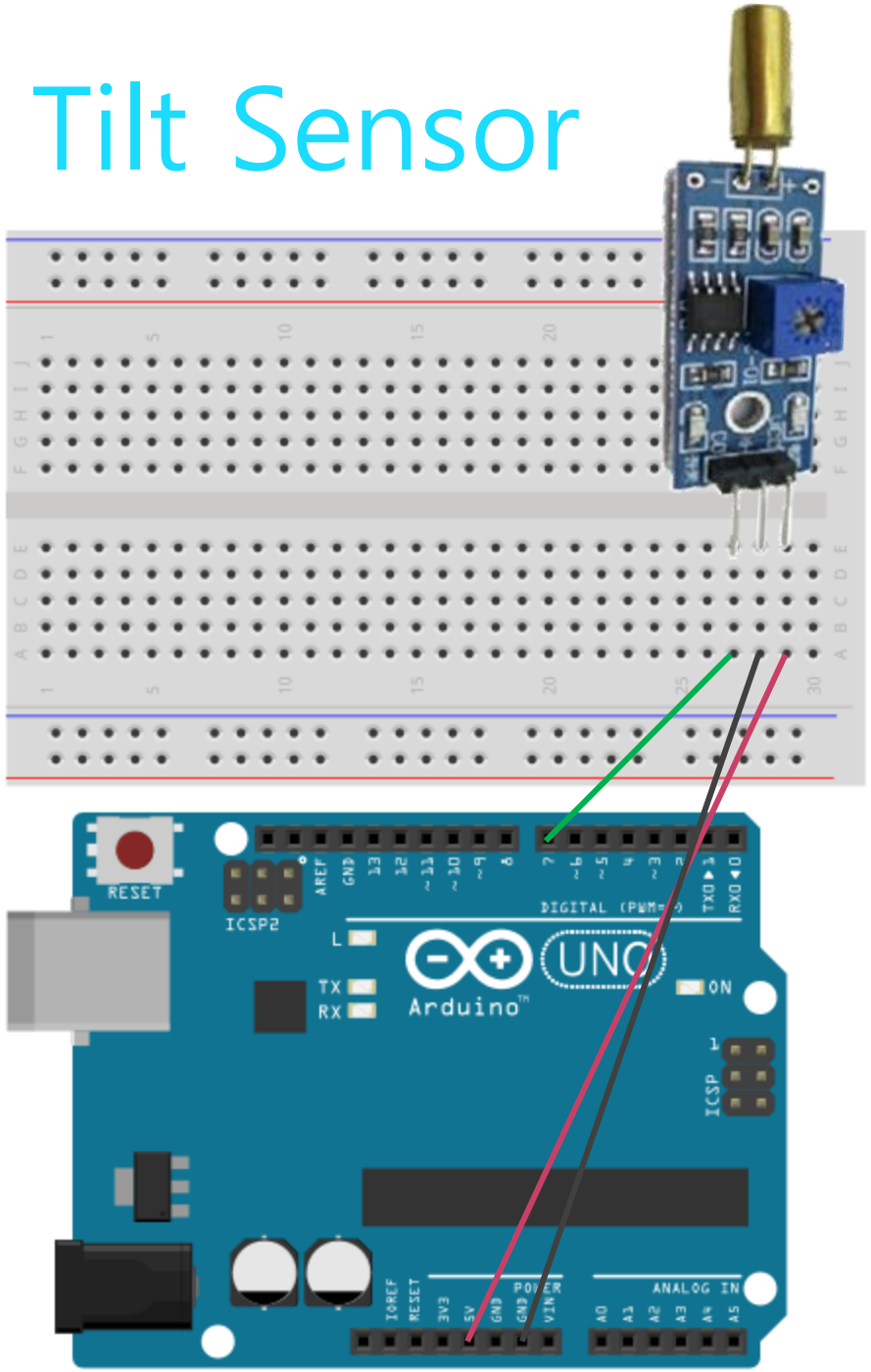
Laser Diode



- 소스코드

```
#define LASER  7

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

void loop() {
    digitalWrite(LASER, HIGH);
    delay(2000);
    digitalWrite(LASER, LOW);
    delay(2000);
}
```

Tilt Sensor



- 소스코드

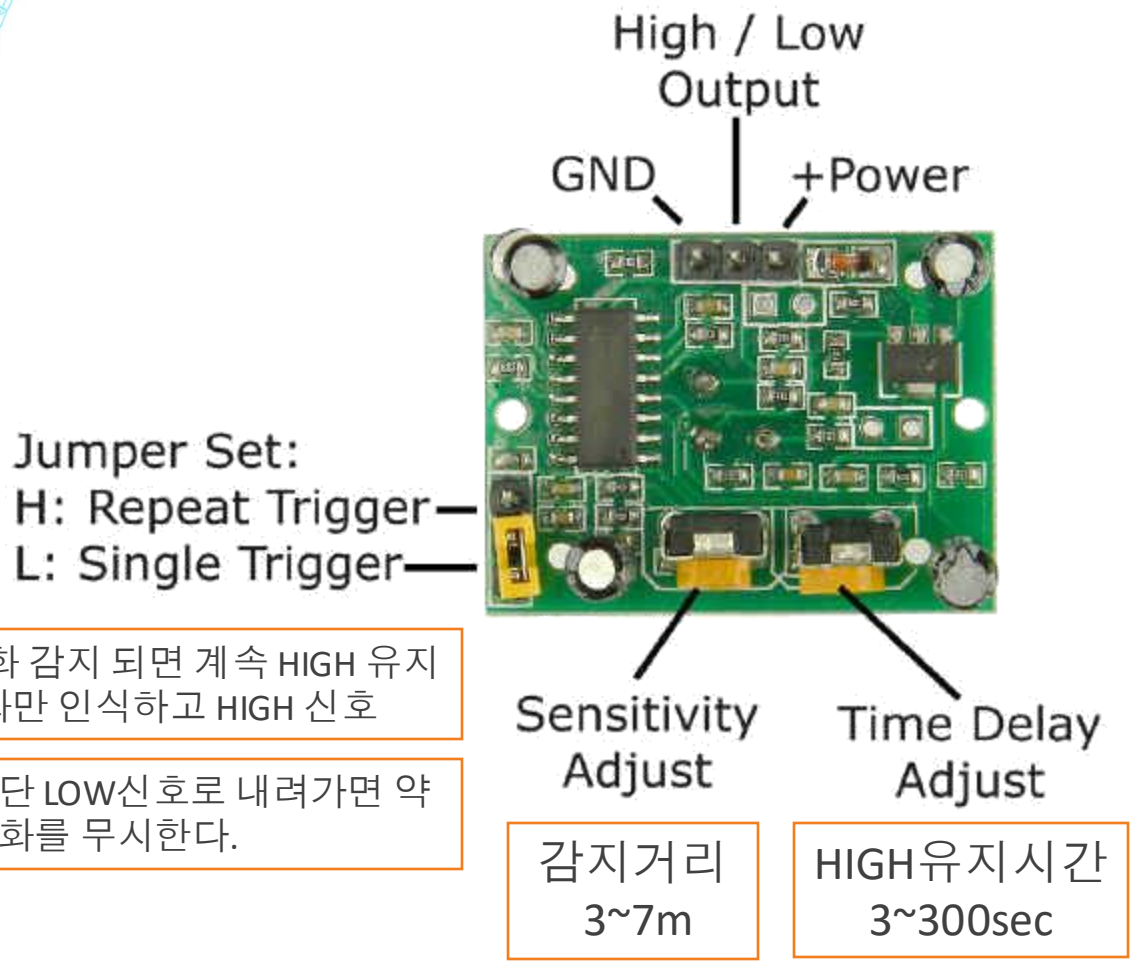
```
#define TILT  7

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

void loop() {
    bool r = digitalRead(TILT);
    Serial.println(r, DEC);
    delay(300);
}
```

Pyroelectric, IR motion 센서

- 모션 감지 센서
  - 적외선에 민감한 특수 물질로 만들어진 센서를 포함하고 있으며 외부에서 들어오는 적외선을 두 개의 영역에서 감지
  - 이 두 영역에서 감지되는 적외선의 변화를 파악해서 모션을 감지하는 원리
  - 적외선의 '변화'를 감지하기 때문에 사람이 감지된 상태라도 그 상태 그대로 가만히 있을 경우 변화 없음 시그널 출력

Pyroelectric, IR motion 센서

- 센서 설정



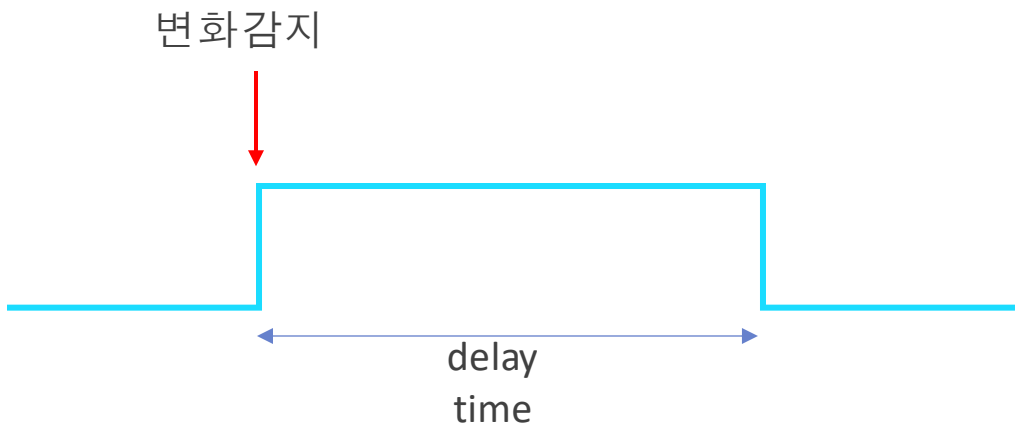
H: 계속 변화 감지 되면 계속 HIGH 유지
L: 최초 변화만 인식하고 HIGH 신호

※ 주의 : 일단 LOW신호로 내려가면 약 5초간은 변화를 무시한다.

감지거리
3~7m

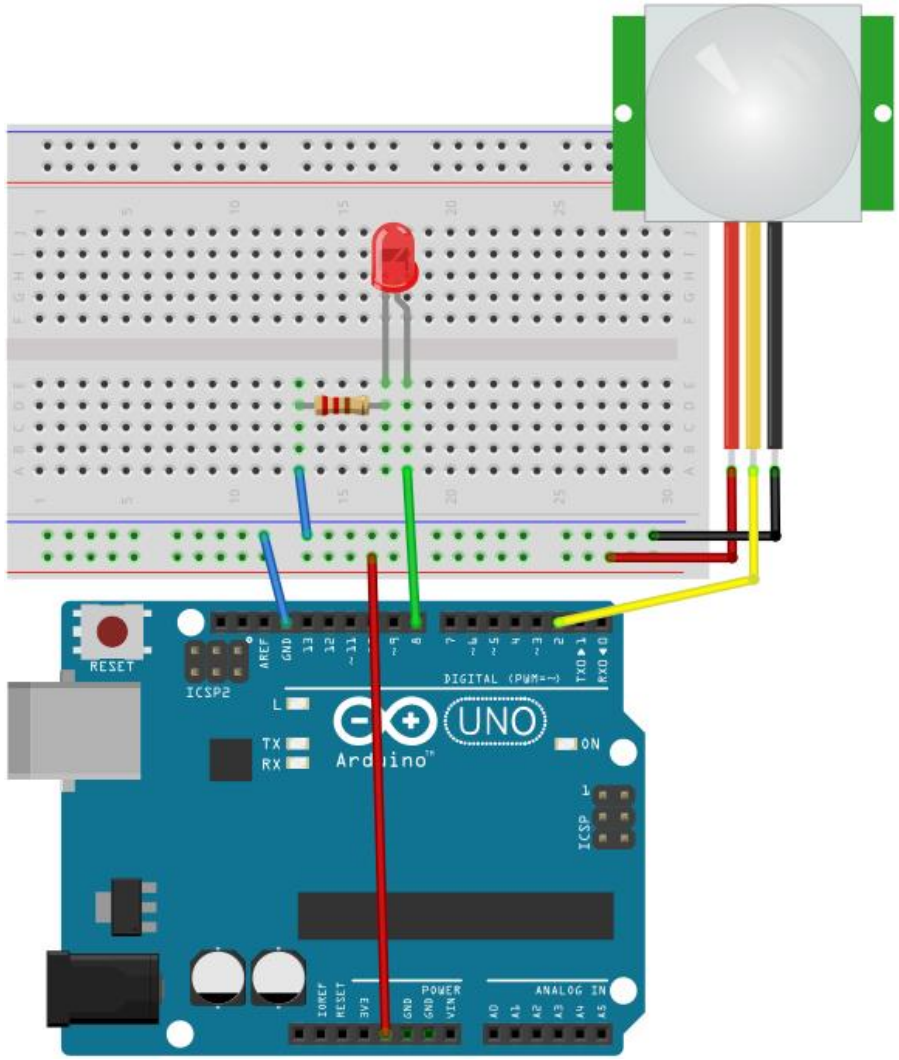
HIGH유지시간
3~300sec

- Retriggering setting jumper
  - L: single trigger
  - H: continuous signal
- <-소 105 대->
- Timing Diagram



Pyroelectric, IR motion 센서

- 회로 결선

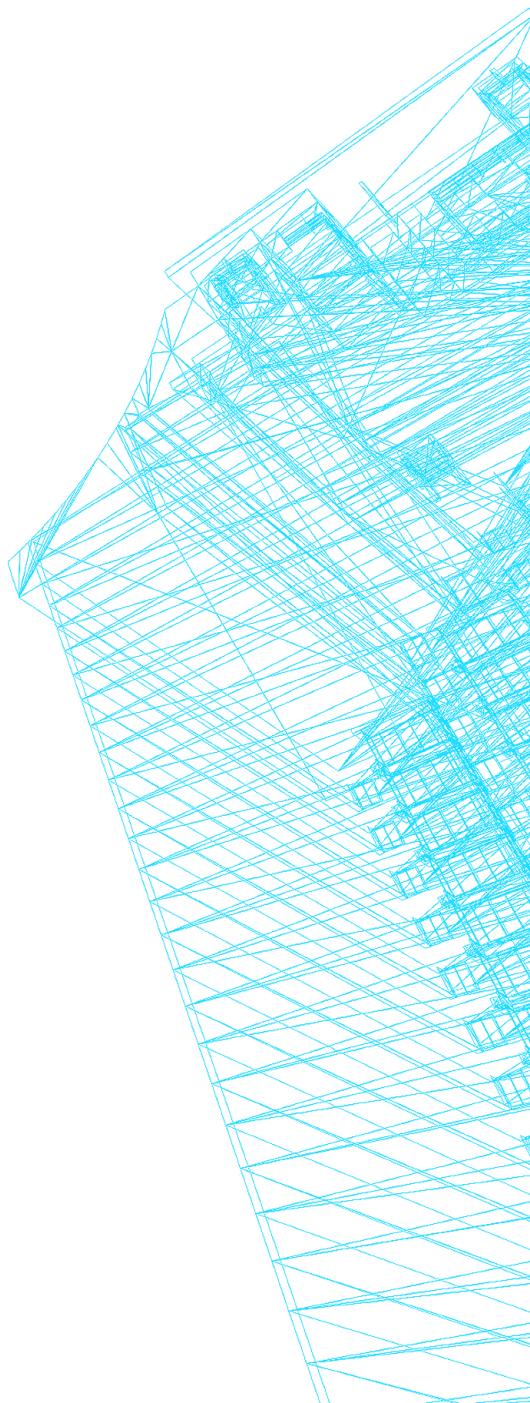


- 소스코드

```
#define PIR 2
#define LED 8

void setup() {
  pinMode(PIR, INPUT);
  pinMode(LED, OUTPUT);
}

void loop() {
  bool b = digitalRead(PIR);
  digitalWrite(LED, b);
}
```

Analog Output

- 펄스 폭 변조(PWM)
- LED 아날로그 출력
  - LED fade in / fade out
- 음과 주파수
  - 피에조 부저 출력

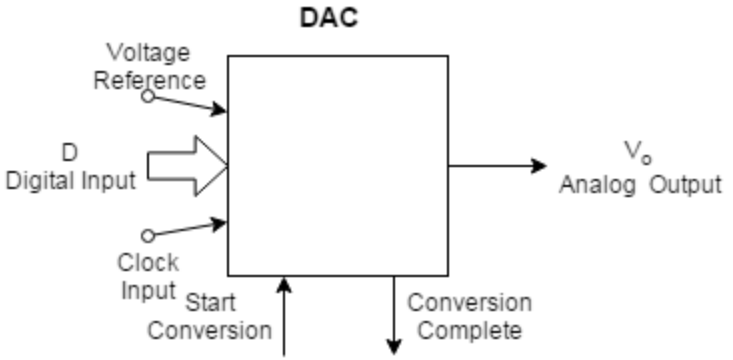


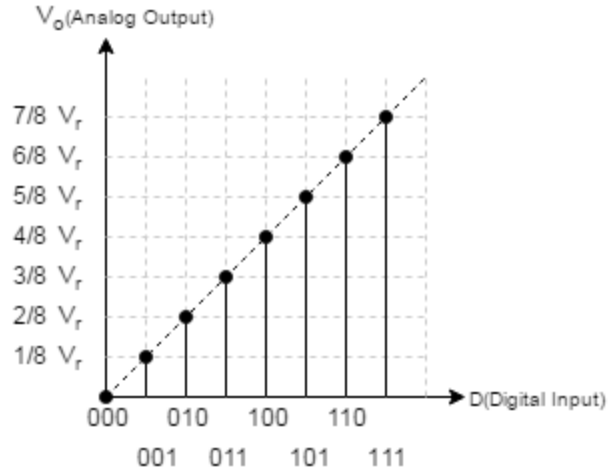
아날로그 출력의 개념

- 출력을 단계별로 조절할 수 없을까?

0%	0.00V	
25%	1.25V	
50%	2.50V	
75%	3.75V	
100%	5.00V	

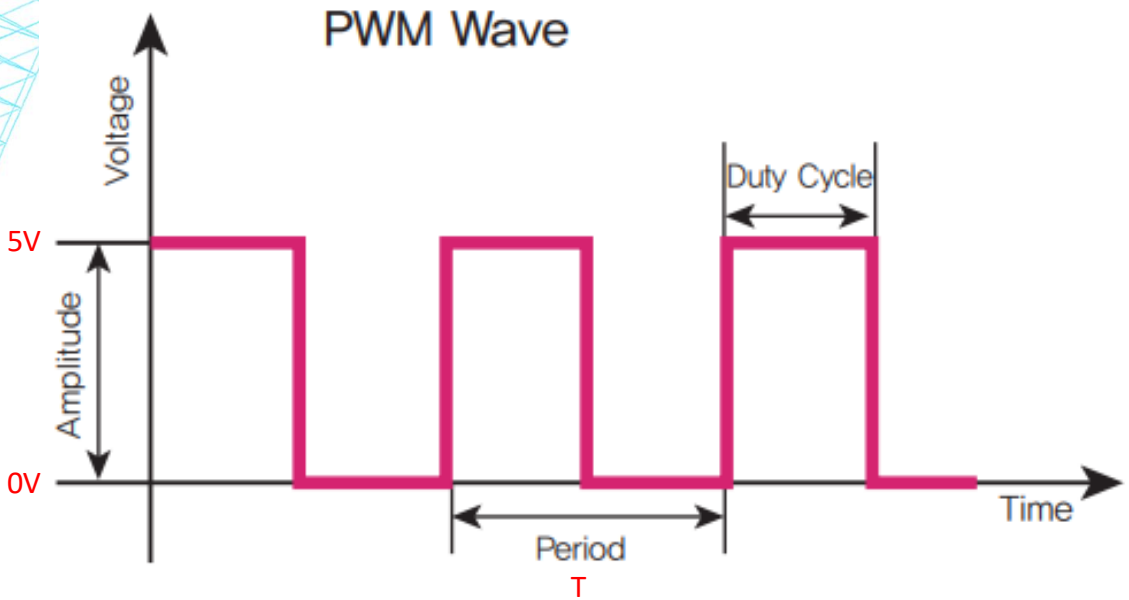
- 최선의 답은 DAC (Digital to Analog Converter) 장치



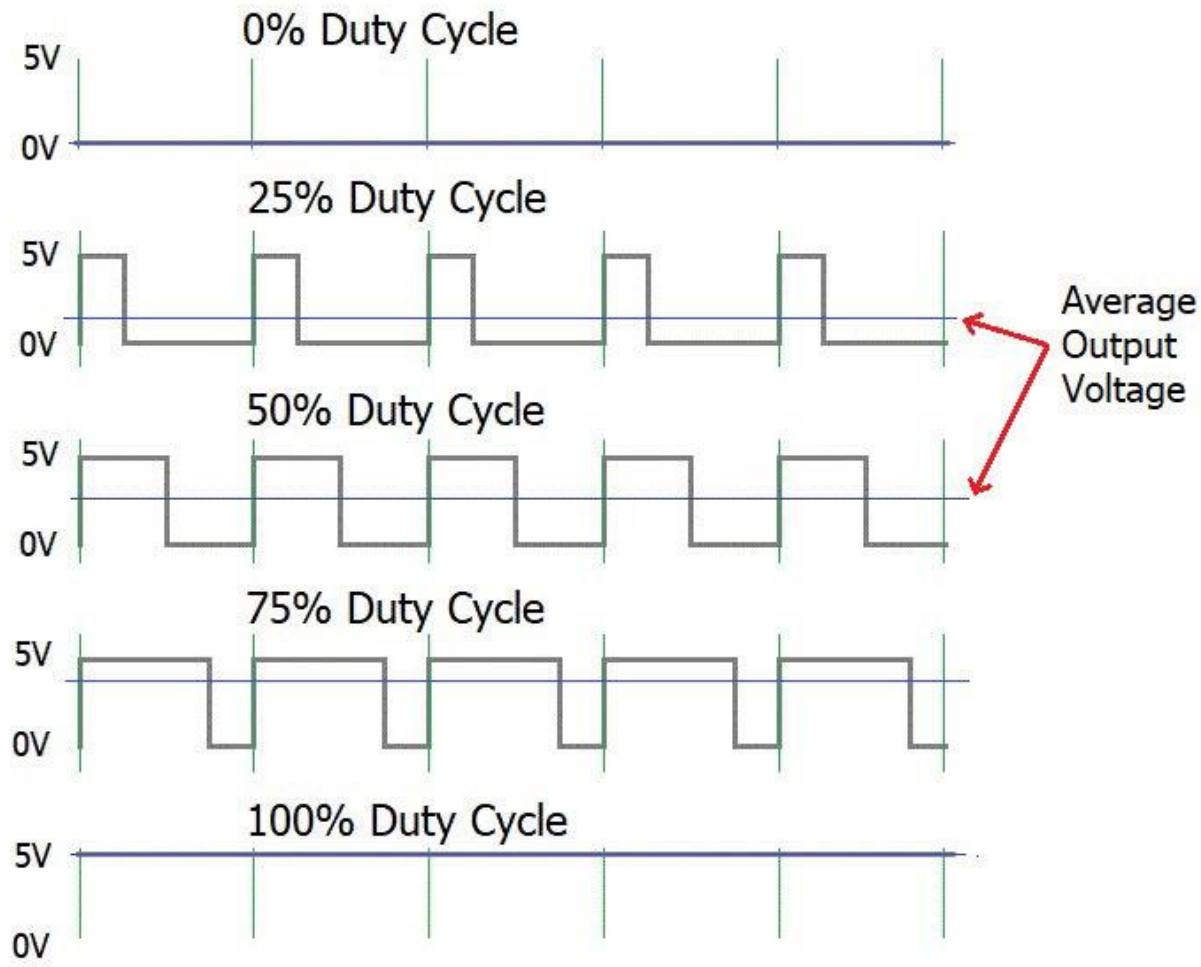


펄스 폭 변조
(PWM, Pulse Width Modulation)

PWM: 펄스폭을 조절하는 디지털 출력으로 아날로그 출력을 흉내내는 기교



▲ 시간에 따른 디지털 출력 변화 표현



펄스 폭 변조
(PWM, Pulse Width Modulation)

$$T = \frac{1}{f}$$

$$f = \frac{1}{T}$$

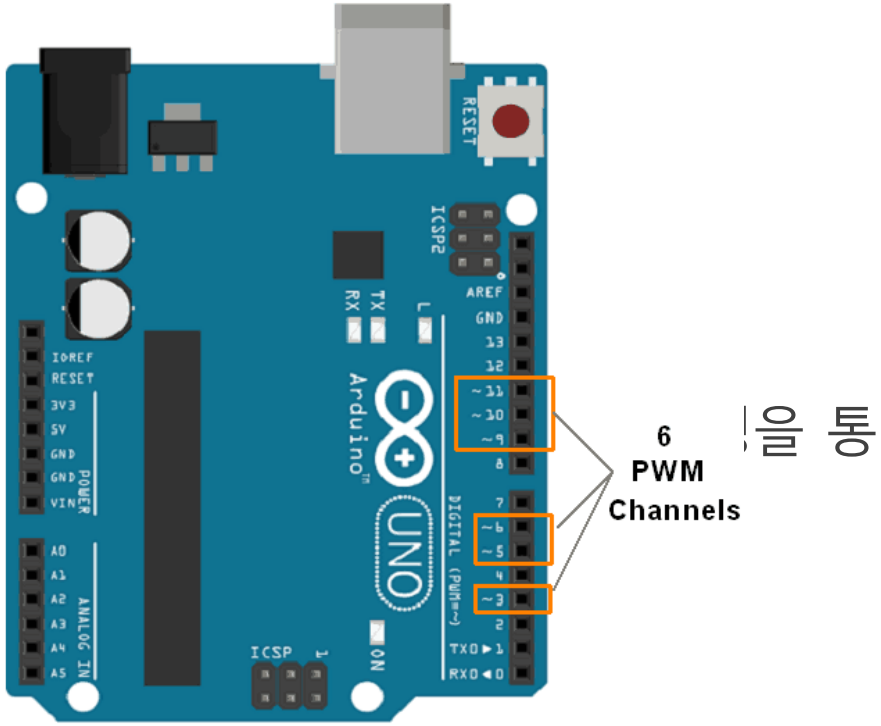
- 아두이노 UNO의 PWM

핀번호	PWM주파수	주기
3,9,10,11	490Hz	0.00204초
5, 6	980Hz	0.00102초

- 아두이노 MEGA의 PWM

핀번호	PWM주파수	주기
2-13, 44-46	490Hz	0.00204초
4, 13	980Hz	0.00102초

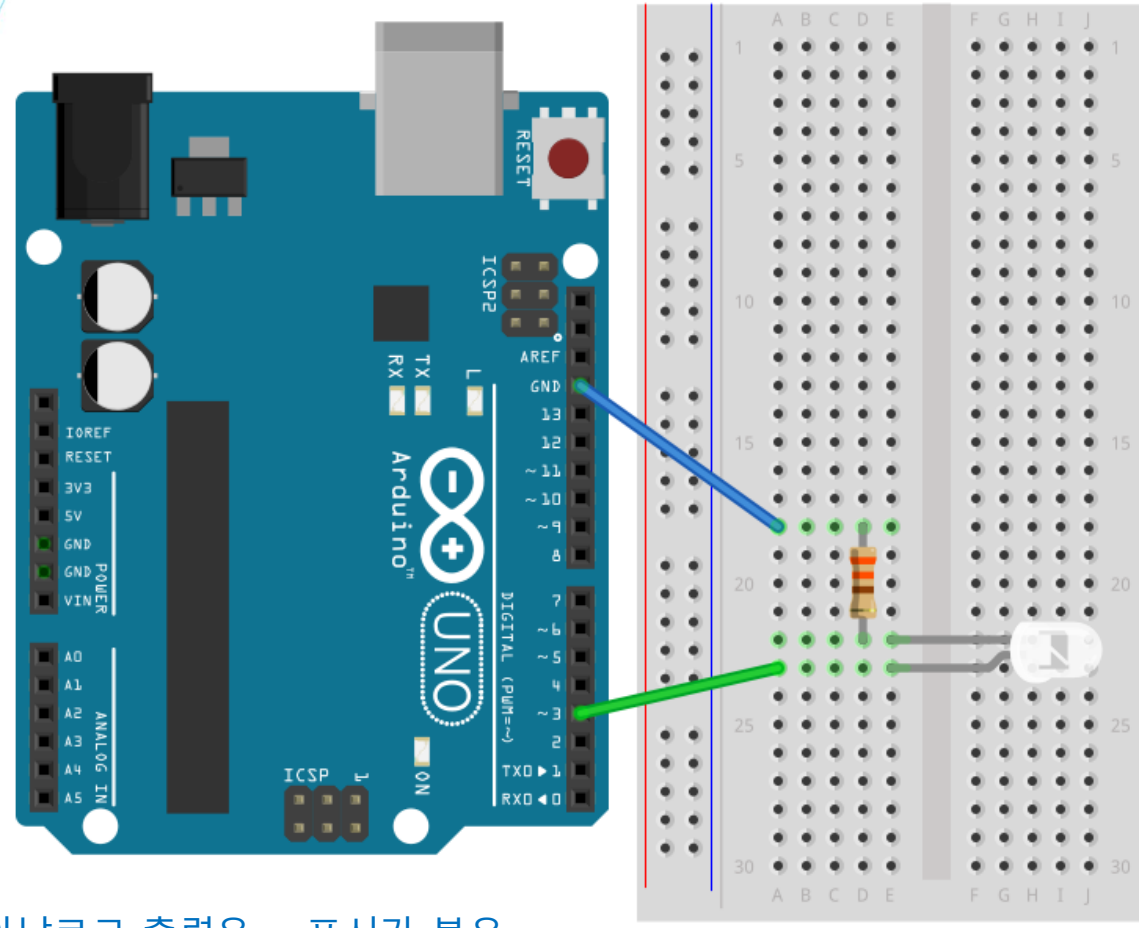
- 수동으로 주파수 변경 가능
  - 참고: https://m.blog.naver.com/PostView.nhn?blogId=schmacher&logNo=221405989919



을 통

LED 페이드 인 / 페이드 아웃

- 회로구성



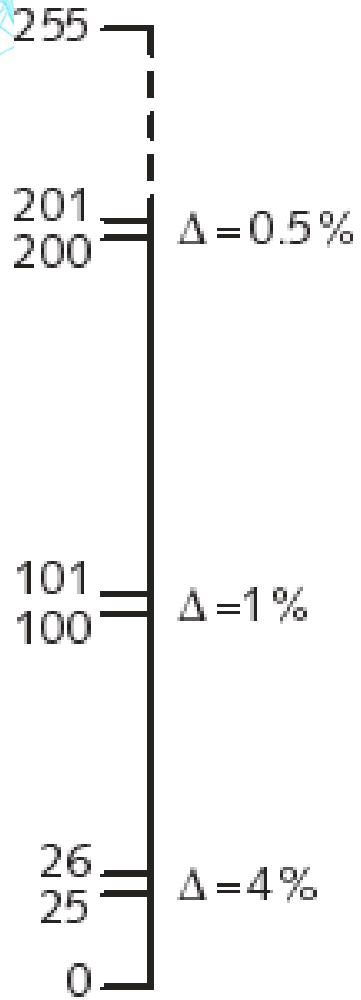
아날로그 출력은 ~ 표시가 붙은
3, 5, 6, 9, 10, 11번 핀을 사용하는 것에 주목

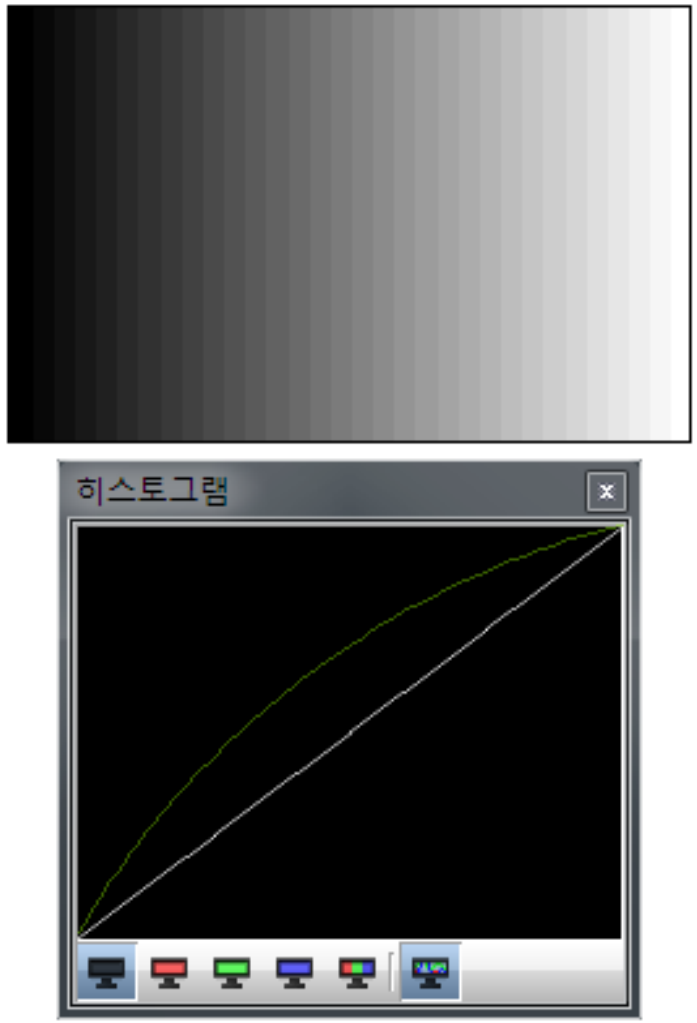
- 소스코드

4-1.ino

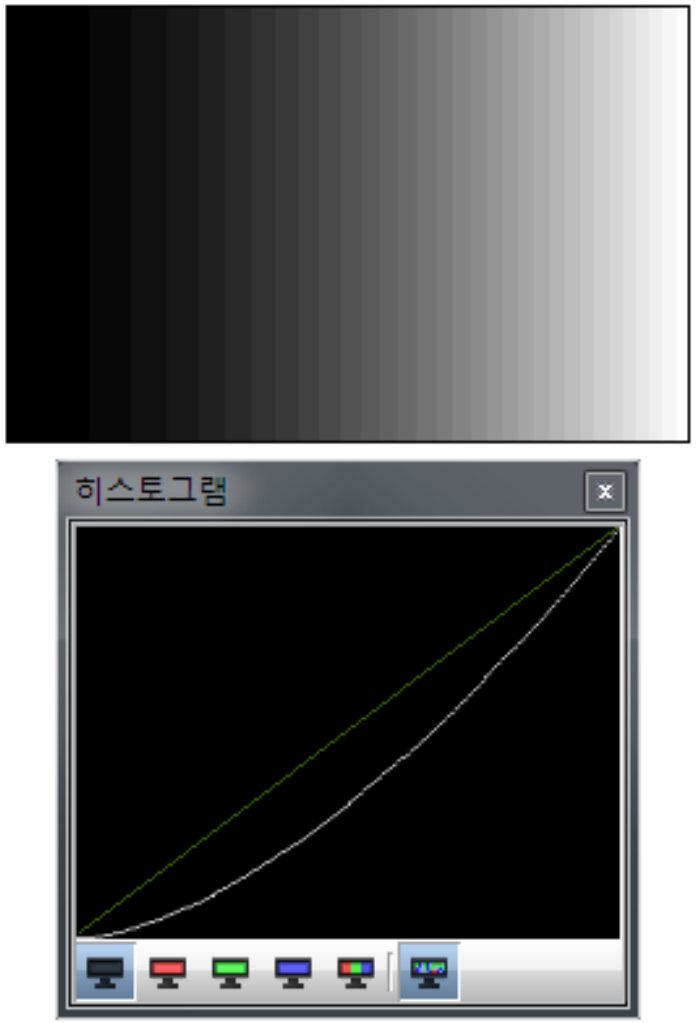
```
#define LED 3

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

void loop() {
      int i;
      // 페이드 인: 서서히 밝히기
      for(i=0; i<256; i+=4) {
            analogWrite(LED, i);
            delay(50);
      }
      // 페이드 아웃: 서서히 끄기
      for(i=255; i>=0; i-=4) {
            analogWrite(LED, i);
            delay(50);
      }
}
```

Gamma correction





a. Linear-wedge gray-scale image.



b. Gamma corrected wedge

Gamma correction

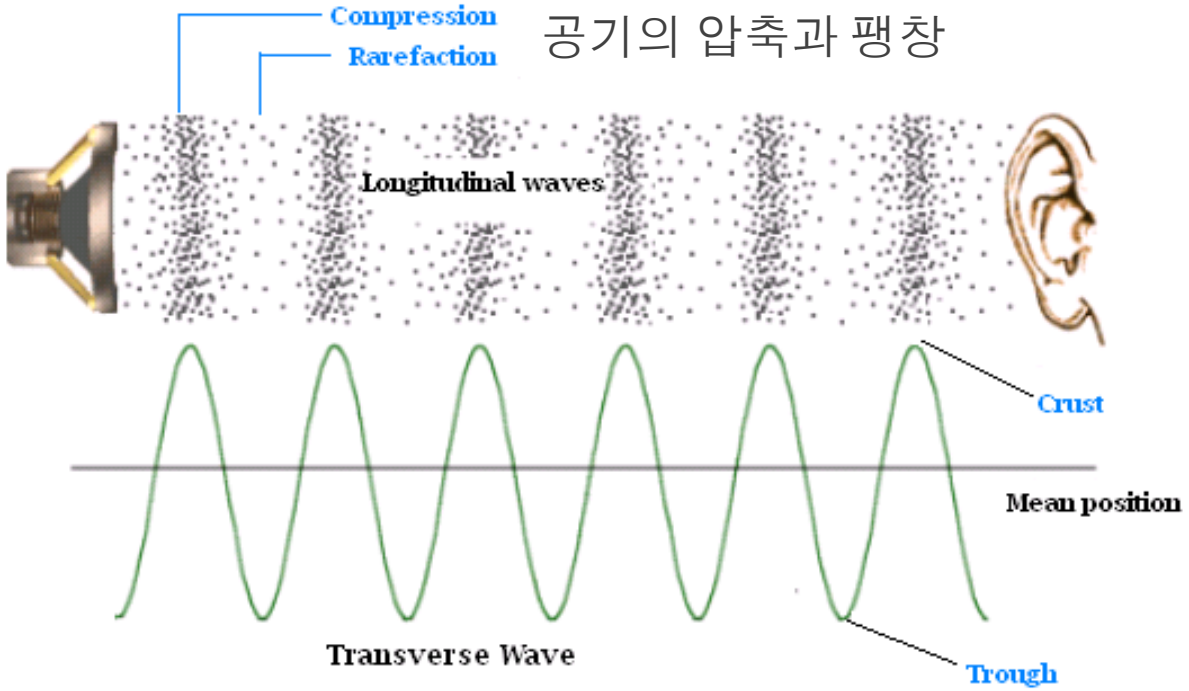
```
class Program
{
    static void Main(string[] args)
    {
        // Generate an LED gamma-correction table for Arduino sketches.
        // Copy-and-paste the program's output into an Arduino sketch.

        double gamma  = 2.2;    // Correction factor
        int    max_in  = 64;    // Top end of INPUT range
        int    max_out = 255;   // Top end of OUTPUT range

        Console.WriteLine("const uint8_t gamma[] = {");
        for(int i=0; i<=max_in; i++) {
            if (i > 0)
                Console.Write(',');
            if ((i != 0) && (i % 16) == 0)
                Console.Write("\n ");

            int v = (int)(Math.Pow((float)i / (float)max_in, gamma) * max_out + 0.5);
            Console.Write(String.Format("{0,3}", v));
        }
        Console.WriteLine(" };");
    }
}
```

Gamma correction

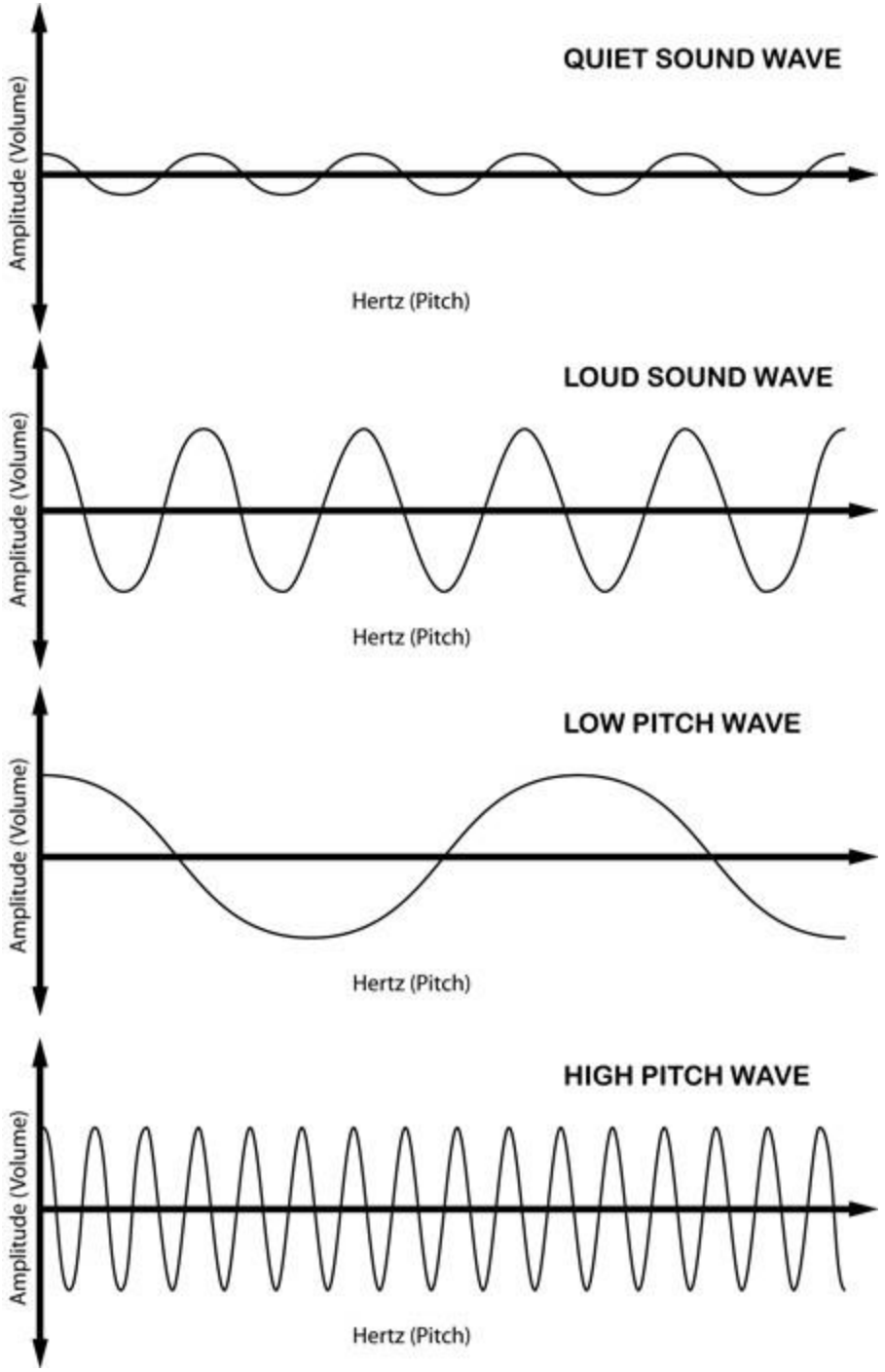
```
const uint8_t gamma[] = {
     0,   0,   0,   0,   1,   1,   1,   2,   3,   4,   4,   5,   7,   8,   9,  11,
    13,  14,  16,  18,  20,  23,  25,  28,  31,  33,  36,  40,  43,  46,  50,  54,
    57,  61,  66,  70,  74,  79,  84,  89,  94,  99, 105, 110, 116, 122, 128, 134,
   140, 147, 153, 160, 167, 174, 182, 189, 197, 205, 213, 221, 229, 238, 246, 255
};
#define LED 3

void setup() {
      pinMode(3,OUTPUT);
}

void loop() {
      int i;
      for(i=0; i<64; i++) {
            int c = gamma[i];
            analogWrite(3, c);
            delay(30);
      }
      for(i=63; i>0; i-=1) {
            int c = gamma[i];
            analogWrite(3, c);
            delay(30);
      }
}
```

음과 주파수

- 음파

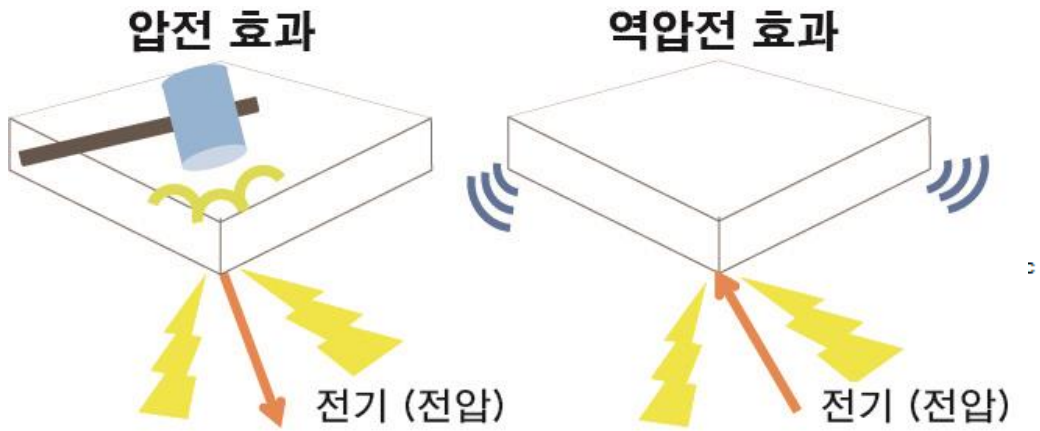


공기의 압축과 팽창

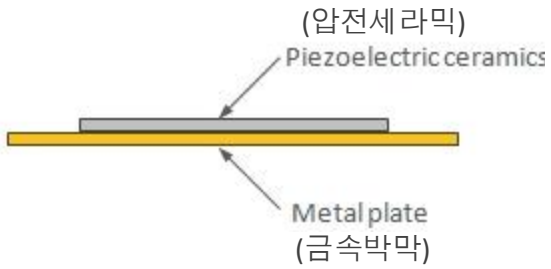


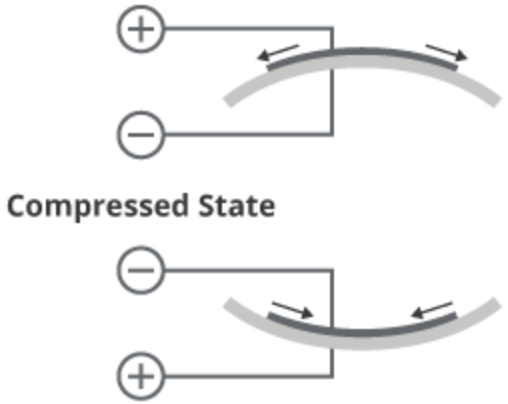
소리의 크고 작음은 진폭에 의해서,
소리의 높고 낮음은 주파수에 의해서 결정

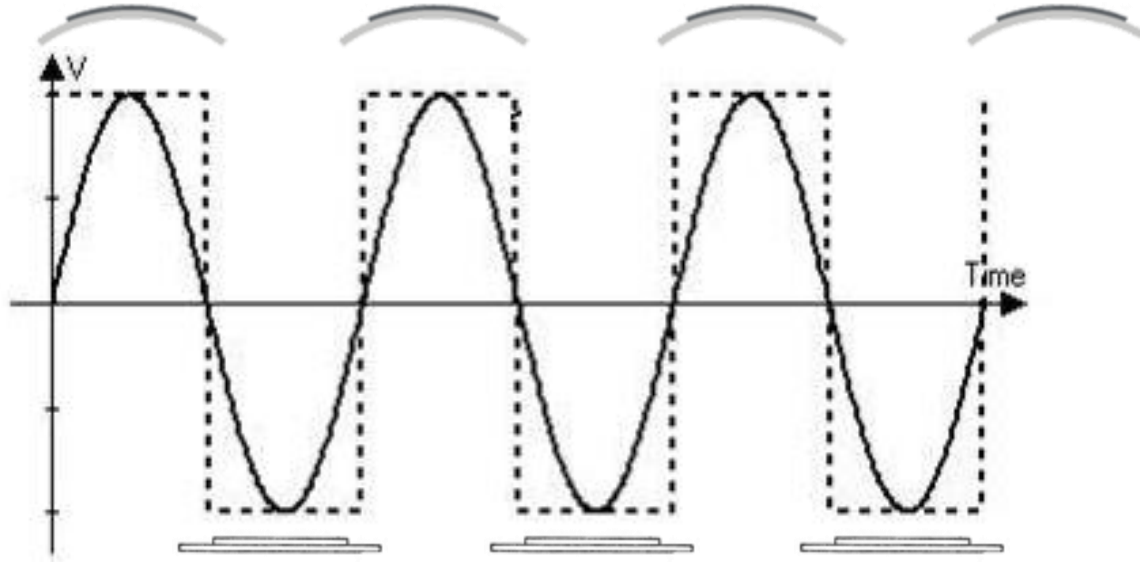
피에조 부저


압전 효과
역압전 효과
전기 (전압)
전기 (전압)

【피에조 효과】


(압전세라믹)
Piezoelectric ceramics
Metal plate
(금속박막)


Compressed State


V
Time

피에조 부저

- 능동 부저
  - 전압이 걸리면 내부 발진기를 사용하여 미리 정해 놓은 소리가 만들어진다.

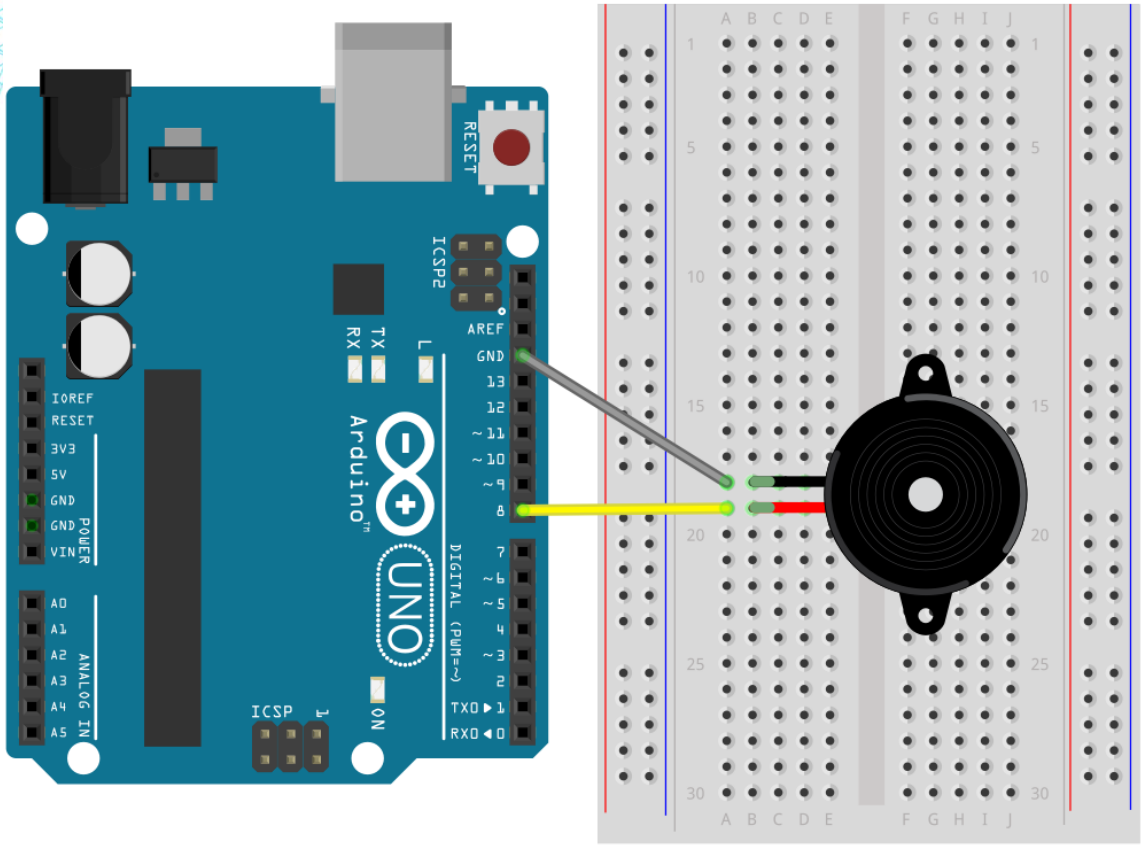
- 수동 부저
  - 내부 발진기가 없으므로 직접 주파수를 발생시켜 소리를 만들어 내야 한다.

tone() 함수

- **tone(pin, freq [, duration])**
  - pin : 부저나 스피커가 연결된 디지털 핀 번호
  - freq : 주파수 (범위 : 31 ~ 65535), 31보다 작은 값을 넣어주면 잡음 출력
  - duration : (옵션) 음의 발생 시간 (밀리초)
- 핀에 지정된 주파수(50 % 듀티비)의 구형파(矩形波, square wave)를 발생시킨다. 지속 시간을 정할 수 있으며 따로 지정하지 않으면 noTone()을 호출 할 때까지 계속 된다.
- 핀은 피에조 부저 또는 스피커에 연결하여 톤을 재생할 수 있다.
- 한번에 하나의 톤 만 생성 할 수 있다. tone()이 이미 다른 핀에서 재생 중이면 tone() 호출은 아무 효과가 없다.
- Mega/ADK 이외의 보드에서 tone() 함수를 사용하면 3번과 11번 핀의 PWM 출력을 방해한다(즉, analogWrite 못쓴다는 뜻)
- tone() 함수는 기능을 수행하기 위해 하드웨어 PWM이 아닌 마이크로컨트롤러의 timer 사용한다. 따라서 디지털 출력을 할 수 있는 어떤 핀에든 할당 가능하다.

Passive buzzer

- 회로구성



- 소스코드

tone.ino

```
#define LED 13
#define BUZZER 8

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

void loop() {
  tone(BUZZER, 440); // 440Hz 소리내고
  delay(500);        // 0.5초
  noTone(BUZZER);    // 소리끄고
  delay(500);        // 0.5초

  digitalWrite(LED, HIGH);
  tone(BUZZER, 440, 500);
  digitalWrite(LED, LOW);
  delay(1000);
}
```

그 밖의 전자음

- Warning 사운드

4-4.ino

```
#define BUZZER 8

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

void loop() {
    tone(BUZZER, 600);
    delay(600);

    tone(BUZZER, 300);
    delay(600);
}
```

- Falling 사운드

4-5.ino

```
#define BUZZER 8

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

void loop() {
  for(int f=20; f<5000; f+=20) {
    tone(BUZZER, f);
    delay(20);
  }
  for(int f=5000; f>20; f-=20) {
    tone(BUZZER, f);
    delay(20);
  }
}
```

그 밖의 전자음

- 외계 사운드

4-6.ino

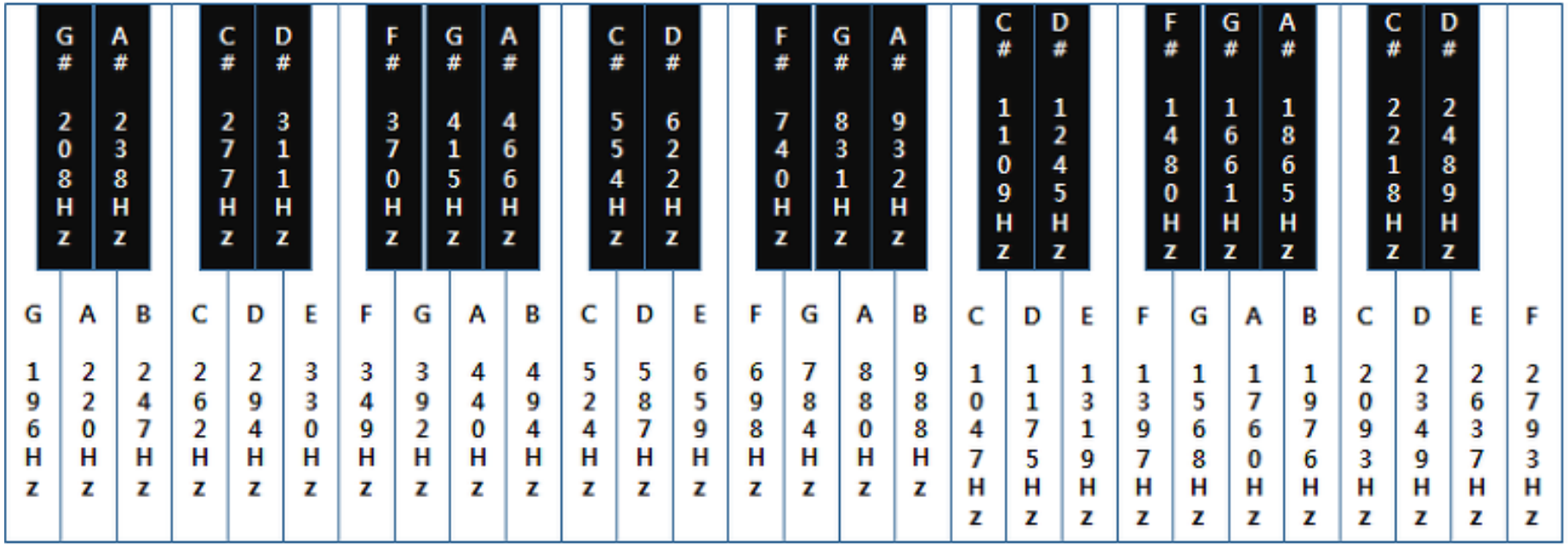
```
#define BUZZER 8

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

void loop() {
   // 외계의 소리
   int r = rand() % 5000;
   tone(BUZZER, r+20);
   delay(100);
}
```

- 폭발

```
#define BUZZER 8

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

void loop() {
 // 떨어지는 소리
  for(int f=5000; f>20; f-=20) {
    tone(BUZZER, f);
    delay(20);
  }
  // 폭발 소리
  for(int i=0; i<300; i++) {
    int r=rand()%120+40;
    tone(BUZZER, r);
    delay(50);
  }
}
```

음과 주파수

- 주파수 구성요소
  - 주파수가 2배인 음의 폭 = 옥타브
  - 4옥타브 라 = 440Hz



음과 주파수

- 음계별 주파수

(단위 : Hz)

옥타브 / 음계	1	2	3	4	5	6	7	8
C(도)	32.7032	65.4064	130.8128	261.6256	523.2511	1046.502	2093.005	4186.009
C#	34.6478	69.2957	138.5913	277.1826	554.3653	1108.731	2217.461	4434.922
D(레)	36.7081	73.4162	146.8324	293.6648	587.3295	1174.659	2349.318	4698.636
D#	38.8909	77.7817	155.5635	311.1270	622.2540	1244.508	2489.016	4978.032
E(미)	41.2034	82.4069	164.8138	329.6276	659.2551	1318.510	2637.020	5274.041
F(파)	43.6535	87.3071	174.6141	349.2282	698.4565	1396.913	2793.826	5587.652
F#	46.2493	92.4986	184.9972	369.9944	739.9888	1479.978	2959.955	5919.911
G(솔)	48.9994	97.9989	195.9977	391.9954	783.9909	1567.982	3135.963	6271.927
G#	51.9130	103.8262	207.6523	415.3047	830.6094	1661.219	3322.438	6644.875
A(라)	55.0000	110.0000	220.0000	440.0000	880.0000	1760.000	3520.000	7040.000
A#	58.2705	116.5409	233.0819	466.1638	932.3275	1864.655	3729.310	7458.620
B(시)	61.7354	123.4708	246.9417	493.8833	987.7666	1975.533	3951.066	7902.133

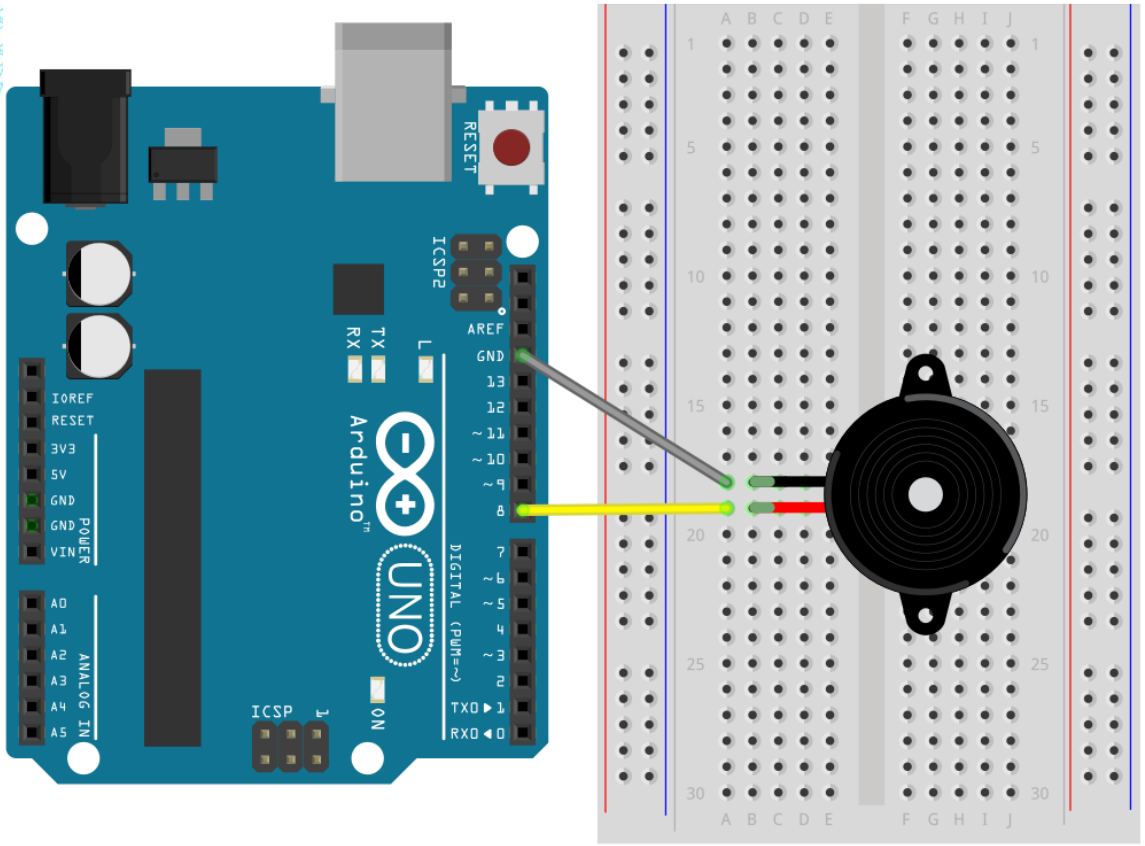
음과 주파수

- 음계의 주파수 구성요소
  - 4옥타브 라 = 440Hz
  - 주파수가 2배인 음의 폭 = 옥타브
  - 총 12음계 존재 (라,라#,시,도,도#,레,레#,미,파,파#,솔,솔#,라)
  - 옥타브 구간의 주파수를 동일한 간격으로 12분할하여 12음계에 배당

옥타브	음계	주파수	계산
4	라	440	$440 * 2^{0/12}$
4	라#	466	$440 * 2^{1/12}$
4	시	494	$440 * 2^{2/12}$
4	도	523	$440 * 2^{3/12}$
4	도#	554	$440 * 2^{4/12}$
4	레	587	$440 * 2^{5/12}$
4	레#	622	$440 * 2^{6/12}$
4	미	659	$440 * 2^{7/12}$
4	파	698	$440 * 2^{8/12}$
4	파#	740	$440 * 2^{9/12}$
4	솔	784	$440 * 2^{10/12}$
4	솔#	831	$440 * 2^{11/12}$
5	라	880	$440 * 2^{12/12}$

음와 주파수

- 회로구성



4-7-1a.ino

```
#define BUZZER 8
#define MAX 48
unsigned int freqz[MAX];
bool isHalf[MAX] =
{0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0,1,0,1,0,0,1,0,1,
 0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0,1,0,1,0,0,1,0,1};

void setup() {
      pinMode(BUZZER, OUTPUT);
      pinMode(13, OUTPUT);
      Serial.begin(9600);
      for(int i=0; i<MAX; i++)  // 음계 생성
            freqz[i] = 220.0 * pow(2, i/12.0);
      for(int i=0; i<MAX; i++)  // 생성된 주파수 값 출력
            Serial.println(freqz[i], DEC);
}

void loop() {
  for(int i=0; i<MAX; i++) {
    if(isHalf[i]) continue; // 반음계는 건너뛰기

    int duration = 500;      // 500ms
    tone(BUZZER, freqz[i]);
    delay(duration);
  }
}
```

Buzzer (Melody 출력)

dingdong3.ino

```
// 딩동, 땡~ 멜로디 출력 예제

#define BUZZER 8
#define BTN_OK 11
#define BTN_NO 12

typedef struct {
 int pitch;
 int length;
} Note;

// 딩동댕동~ (도미솔도~)
Note noteOK[] = { {523,250}, {659,250},
     {784,250}, {1046,500} };
// 땡~ (미~~~~)
Note noteNO[] = { {659,1000} };

void setup() {
  Serial.begin(9600);
  pinMode(BTN_OK, INPUT_PULLUP);
  pinMode(BTN_NO, INPUT_PULLUP);
}
```

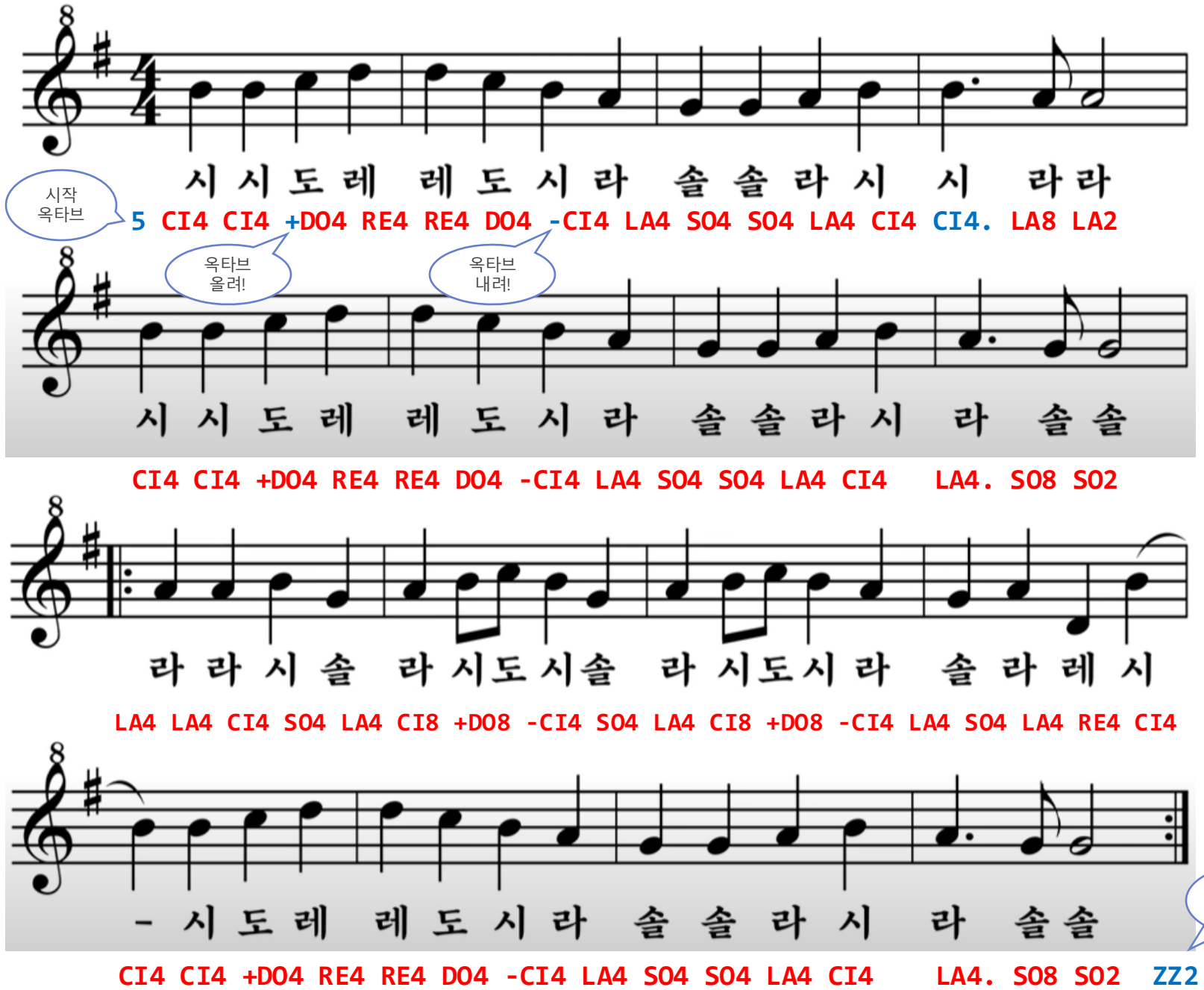
```
void loop() {
  if(digitalRead(BTN_OK)==LOW) {
    int n=sizeof(noteOK)/sizeof(Note); // (12 / 4)
    for(int i=0; i<n; i++) {
      tone(BUZZER, noteOK[i].pitch, noteOK[i].length);
      delay(noteOK[i].length);
    }
  }

  if(digitalRead(BTN_NO)==LOW) {
    int n=sizeof(noteNO)/sizeof(Note); // (4 / 4)
    for(int i=0; i<n; i++) {
      tone(BUZZER, noteNO[i].pitch, noteNO[i].length);
      delay(noteNO[i].length);
    }
  }
}
```

악보를 문자열로 변환 [베토벤 환희의 송가]

[음계표]
도 DO
레 RE
미 MI
파 FA
솔 SO
라 LA
시CI
(쉼표) ZZ

도# DS
레# RS
파# FS
솔# SS
라# LS

레b RF
미b MF
솔b SF
라b LF
시b CF



[사용예]
도4분음표
DO4

레#2분음표쩜
RE#2.

4분음 쉼표
ZZ4

한 옥타브 올려서
미8분음표쩜
+MI8.

한 옥타브 내려서
시8분음표
-CI8

5옥타브라4분음표
5LA4
:
:

문자열로 기록된 악보를 해석하는 CNote 클래스

CNote.h

```
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <math.h>

class CNote {
  private:
    char *_score_origin;   // 악보 문자열 원본
    char *_score;          // 악보(플레이 하면서 오염됨)
    char *_score_head;     // 악보 시작 지점
    int  _tempo;
    bool _auto_replay;
    char *_cnote; // current_note
    const char *_delimiter =" ";
    int _note_octave;
    int _note_pitch;
    int _note_length;

  public:
    CNote();
    void  score(char *str);
    void  tempo(int t);
    bool  next();
    char* note();
    int   pitch();
    int   length();
    int   octave();
    int   freq(char* note);
    void  replay(bool val);
};

CNote::CNote() {
  _auto_replay = true;
  _tempo = 120;
  _score_head = _score = NULL;
  _note_pitch = 0;
  _note_length = 0;
  _note_octave = 0;
}

void CNote::score(char *str) {
  _score_origin = str;

  if(_score_head != NULL)
    free(_score_head);

  _score = malloc(strlen(str)*sizeof(char));
  strcpy(_score, str);
  _score_head = _score;
  _cnote = NULL;
}
```



```
void CNote::tempo(int tempo) {  _tempo = tempo;  }

bool CNote::next() {
  if(_cnote == NULL)   // 처음인 경우
    _cnote = strtok(_score, _delimiter);
  else
    _cnote = strtok(NULL, _delimiter);

  if(_cnote == NULL && _auto_replay == true) {
    score(_score_origin);
    return next();
  }

  if(_cnote != NULL)
    _cnote = strupr(_cnote);
  else
    return false;

  char* note = _cnote;

  if(isdigit(note[0])) {
    _note_octave = note[0]-'0';
    if(! (0 <= _note_octave && _note_octave <=9))
      _note_octave = 0;
    note++;
  }
  else if(note[0]=='-') {
    _note_octave--;
    note++;
  }
  else if(note[0]=='+') {
    _note_octave++;
    note++;
  }

  if(strlen(note) <= 0) {
    next();
    return true;
  }

  _note_pitch = freq(note);
  note += 2;

  int len = atoi(note);
  _note_length = 60.0/_tempo *(4000.0/len);

  if(note[strlen(note)-1]=='.')
    _note_length = _note_length*1.5f;

  return true;
}
```

```
int CNote::freq(char* note) {
  int pitch = 0;
  int nNote = -1;
  switch(note[0]) {
    case 'D': nNote = 3; break;
    case 'R': nNote = 5; break;
    case 'M': nNote = 7; break;
    case 'F': nNote = 8; break;
    case 'S': nNote = 10; break;
    case 'L': nNote = 12; break;
    case 'C': nNote = 14; break;
  }

  if(note[1] =='S')
    nNote++;
  else if(note[1] =='F')
    nNote--;

  if(nNote >= 0) {
    float z = ((octave()-1)*12 + nNote)/12.0;
    pitch = 27.5*pow(2, z)+0.49;  // 27.5 = 0 octave LA
  }
  else
    pitch = 0;

  return pitch;
}

char* CNote::note() { return _cnote;       }

int CNote::length() { return _note_length; }

int CNote::pitch()  { return _note_pitch;  }

int CNote::octave() { return _note_octave; }

void CNote::replay(bool val) {
  _auto_replay = val;
}

// Programmed by akapo@naver.com(박정진)
// 2022.6.24.

/* 음계명
  DO(도) RE(레) MI(미) FA(파) SO(솔) LA(라) CI(시)
    DS(도샵) RS(레샵) FS(파샵) SS(솔샵) LS(라샵)
    RF(레플랫)        SF(솔플렛) 등

  표기법 예시
  3DO4 -> 3옥타브 도4분음표
*/
```

MusicPlayer 클래스 구현

MusicPlayer.h

```
#include "CNote.h"

class MusicPlayer {
  private:
    int   _buzzer;
    bool  _enabled;
    bool  _playing;
    bool  _verbose;
    CNote _cnote;
    unsigned long _noteLength;
    unsigned long _preTime;

  public:
    MusicPlayer() { }
    MusicPlayer(int pin) {
      init(pin);
    }
    MusicPlayer(int pin, char* score, int tempo) {
      init(pin);
      setScore(score);
      setTempo(tempo);
    }
```

```
    void init(int pin) {
      pinMode(pin, OUTPUT);
      _buzzer = pin;
      _noteLength = 0;
      _verbose = false;
    }

    void setScore(char* score) {
      _cnote.score(score);
    }

    void setTempo(int tempo) {
     _cnote.tempo(tempo);
    }

    void setReplay(bool b) {
      _cnote.replay(b);
    }

    void setVerbose(bool b) {
      _verbose = b;
    }
```

```
void play() {
  if(!_enabled) { _playing = false;  return; }

  unsigned long curTime = millis();
  if(curTime - _preTime < _noteLength) return;
  else noTone(_buzzer);

  if(_cnote.next()) {
    int notePitch = _cnote.pitch();
    _noteLength = _cnote.length();

    if(notePitch >= 31) {
      tone(_buzzer, notePitch, _noteLength);
      _playing = true;
      _preTime = curTime;
    }

    if(_verbose && Serial) {
      Serial.print(_cnote.note());
      Serial.print(" ");
      Serial.print(_cnote.pitch(), DEC);
      Serial.print(" ");
      Serial.println(_cnote.length(), DEC);
    }
  }
}
```

```
  void start() {
    _enabled = true;
    _preTime = 0;
    play();
  }

  void setEnable(bool b) {
    _enabled = b;
  }
  void enable()    { _enabled = false; }
  void disable()   { _enabled = true;  }
  bool enabled()   { return _enabled;  }
  bool isPlaying() { return _playing;  }
};
```

MusicPlayer 클래스를 이용한 연주

```
#include "MusicPlayer.h"
#define BUZ       8
#define BTN_PLAY  12

// 떳다 떳다 비행기
const char *AIRPLANE_SCORES =
  "5 mi8. re16 do8 re8 mi8 mi8 mi4 re8 re8 re4 mi8 mi8 mi4 \
   mi8. re16 do8 re8 mi8 mi8 mi4 re8 re8 mi8. re16 do2";

// 베토벤 환희의 송가
const char *BEETHOVEN_SCORES =
  "4 CI4 CI4 +DO4 RE4 RE4 DO4 -CI4 LA4 SO4 SO4 LA4 CI4 CI4. LA8 LA2 \
   CI4 CI4 +DO4 RE4 RE4 DO4 -CI4 LA4 SO4 SO4 LA4 CI4 LA4. SO8 SO2 \
   LA4 LA4 CI4 SO4 LA4 CI8 +DO8 -CI4 SO4 LA4 CI8 +DO8 -CI4 LA4 SO4 LA4 RE4 CI4 \
   CI4 CI4 +DO4 RE4 RE4 DO4 -CI4 LA4 SO4 SO4 LA4 CI4 LA4. SO8 SO2 \
   LA4 LA4 CI4 SO4 LA4 CI8 +DO8 -CI4 SO4 LA4 CI8 +DO8 -CI4 LA4 SO4 LA4 RE4 CI4 \
   CI4 CI4 +DO4 RE4 RE4 DO4 -CI4 LA4 SO4 SO4 LA4 CI4 LA4. SO8 SO2 ZZ2";

const char *BUBBLE_BUBBLE =
  "5 CF8 LA8 SO8. FA16 LA8 SO8 FA8 MF8 SO8 FA8 MF16 RE16 ZZ8 \
   FA4. re16 do16 -cf8 +do8 re8 mf8 do8 re16 mf16 mf8 fa8 \
   fa8 so8 la16 so16 zz8 fa8 fa8 so8 la8 cf8 la8 so8. fa16 la8 so8 fa8 mf8 so8 \
   fa8 mf16 re16 zz8 fa4. re16 do16 -cf8 +do8 re8 mf8 do8 re16 mf16 mf8 fa8";

MusicPlayer bgmPlayer(BUZ);
```

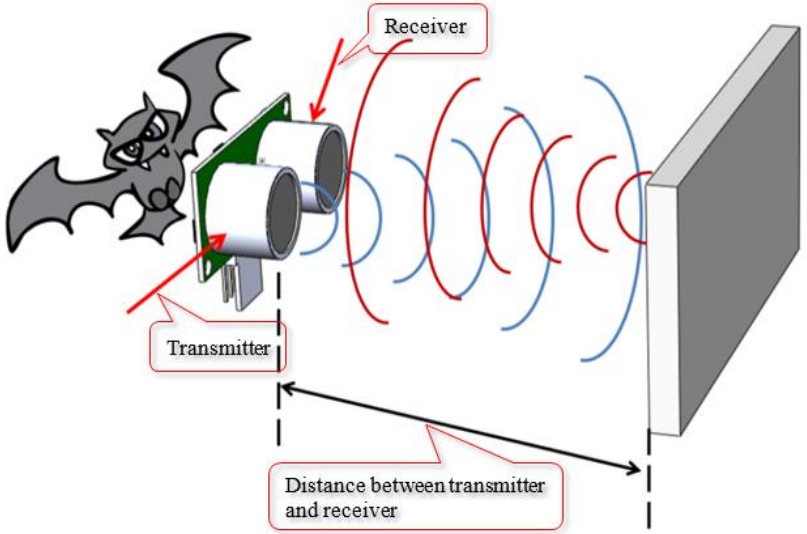

bgmPlay_.ino

```
void setup() {
  Serial.begin(9600);
  pinMode(BTN_PLAY, INPUT_PULLUP);
  bgmPlayer.setScore(BEETHOVEN_SCORES);
  bgmPlayer.setTempo(120);
  bgmPlayer.setVerbose(true);
  bgmPlayer.start();
}

void loop() {
  static bool btn_play_pre = HIGH;
  bool btn_play_now = digitalRead(BTN_PLAY);

  if(btn_play_pre==HIGH && btn_play_now==LOW) {
    bool e = bgmPlayer.enabled();
    bgmPlayer.setEnable(!e);
  }
  bgmPlayer.play();

  btn_play_pre = btn_play_now;
}
```

초음파 센서

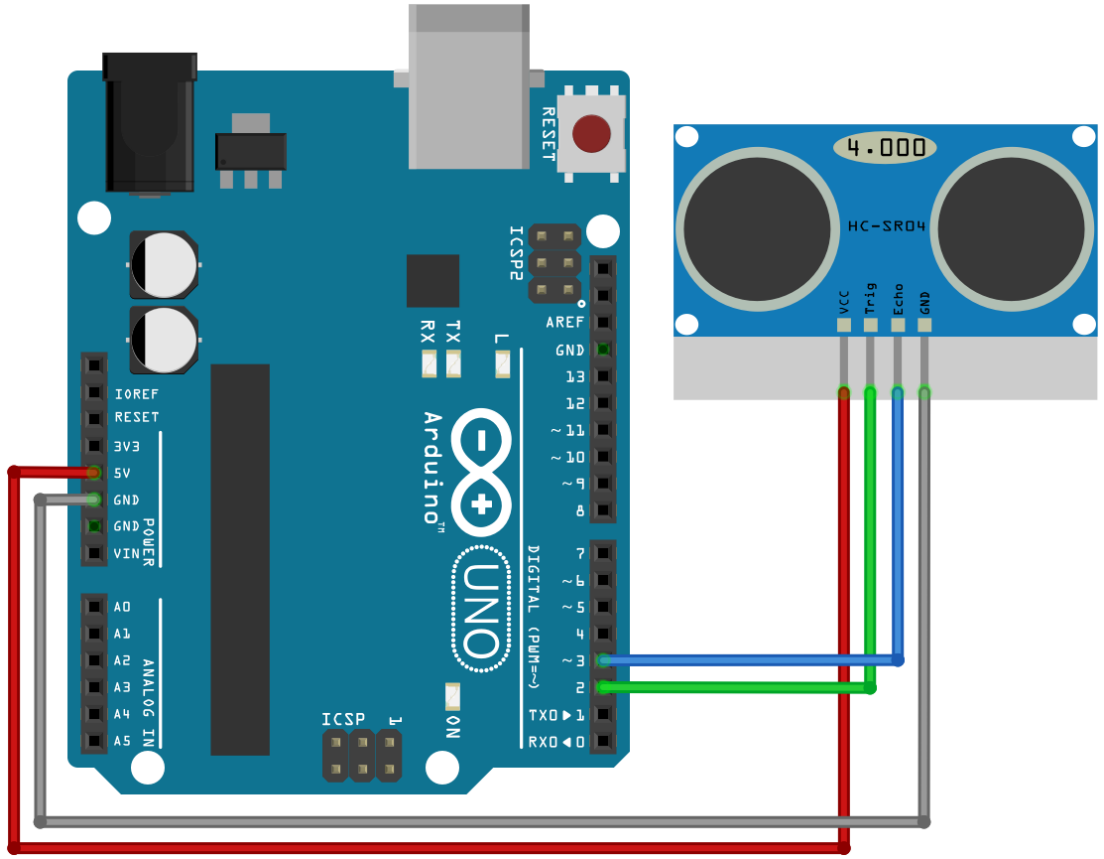
- 외관

- 작동원리



초음파 센서

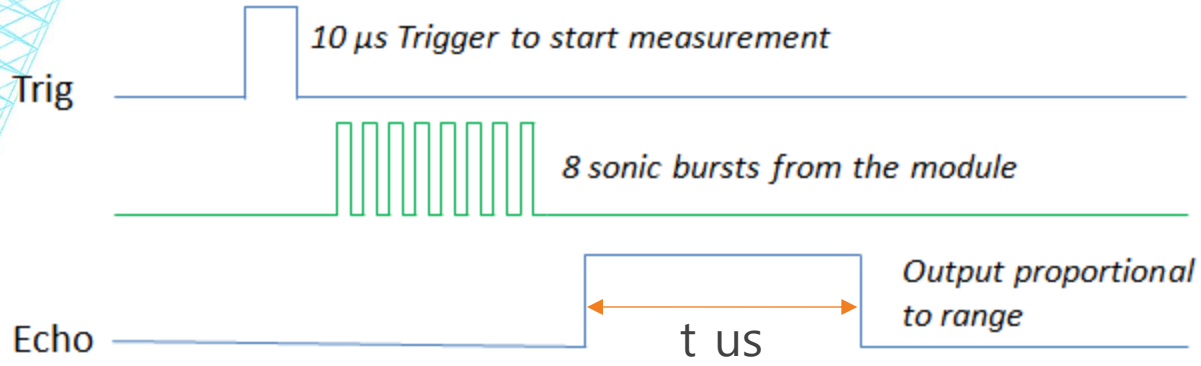
- 회로구성



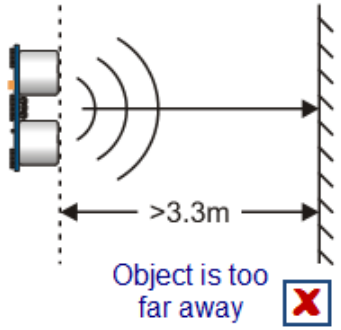

```
#define TRIG 2
#define ECHO 3

void setup() {
  pinMode(TRIG, OUTPUT);
  pinMode(ECHO, INPUT);
  Serial.begin(9600);
}
```

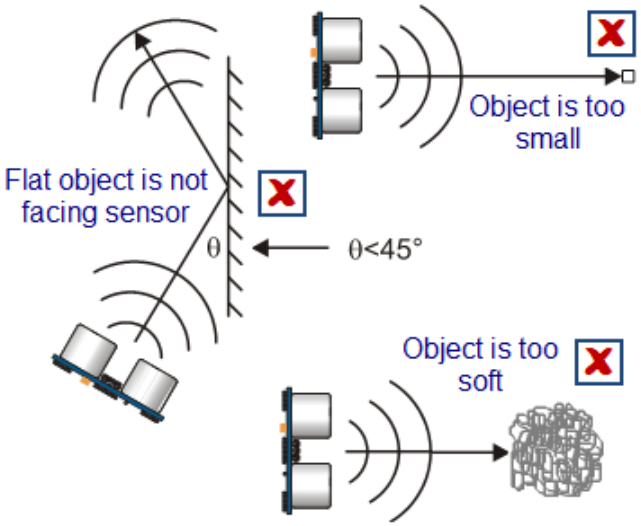
초음파 센서

- 거리 계산 방법



- 감지 한계





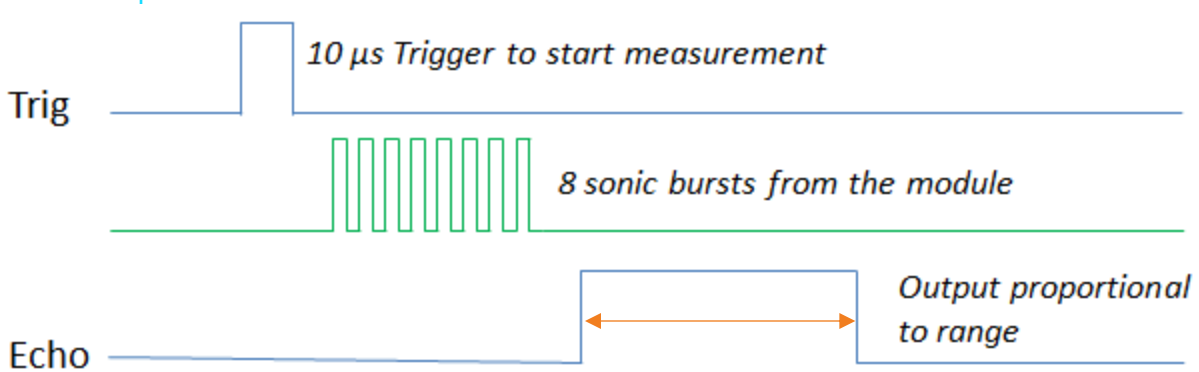
- 음속: 340m/s = 34000cm/s = 0.034cm/us
- v = d / t
- d = v * t
- 2d = 0.034cm x t;
- d = (0.034cm * t) / 2;

초음파 센서

4-7.ino

```
#define TRIG 2
#define ECHO 3

void setup() {
  pinMode(TRIG, OUTPUT);
  pinMode(ECHO, INPUT);
  Serial.begin(9600);
}

void loop() {
  long duration;
  int distanceCm;

  digitalWrite(TRIG, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG, LOW);

  duration = pulseIn(ECHO, HIGH);
  distanceCm= 0.034*duration/2;
  Serial.print(distanceCm);
  Serial.println("cm");

  delay(300);
}
```

- `pulseIn(pin, value)`
  - value가 HIGH이면, pulseIn()은 핀이 HIGH가 되었다가 LOW가 될 때까지의 펄스의 길이를 마이크로초 단위로 반환한다. 단, 타임아웃 내에 완전한 펄스가 수신되지 않은 경우 0을 반환합니다.



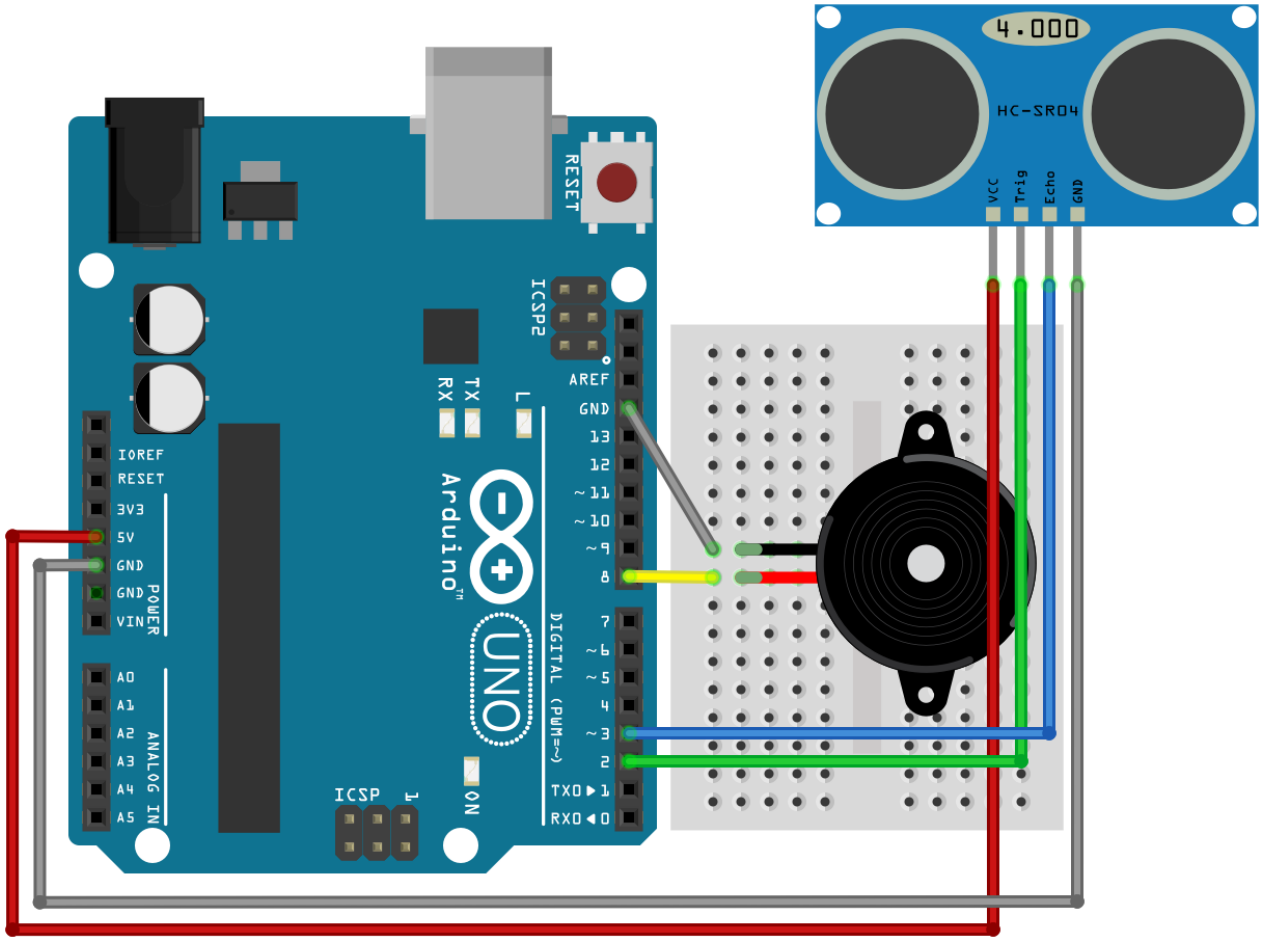
초음파 센서 응용 - 실습과제

- 자동자 후방 감시 센서 구현하기
  - 아두이노, 초음파 센서, BUZZER를 활용하여,
  - 자동차 후방 감지 센서와 같은 역할을 하는 장치를 만드시오.

  - 물체와의 거리가 3미터 이하인 경우에만 소리로 물체까지의 거리를 알려줌
  - 물체와 거리가 가까울 수록 소리 출력의 빈도가 빨라져야 함.

자동차 후방 감지 센서

- 회로 구성



자동차 후방 감지 센서

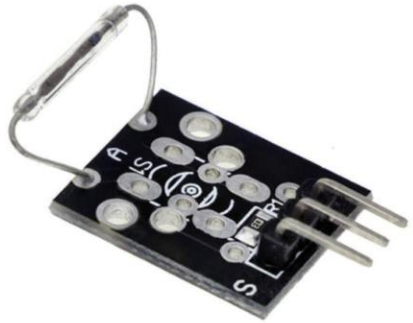
- 소스코드

4-8.ino

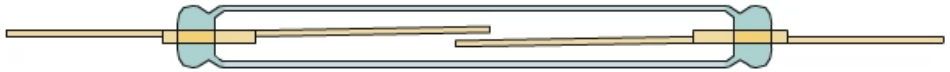
```
// 자동차 후방감지기
// by akapo  (2017.7.3)
#define TRIG 2
#define ECHO 3
#define BUZZER 8

void setup() {
  pinMode(TRIG, OUTPUT);
  pinMode(ECHO, INPUT);
  pinMode(BUZZER, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  long duration;
  int distanceCm;

  digitalWrite(TRIG, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG, LOW);
  duration = pulseIn(ECHO, HIGH);
  distanceCm = duration*0.034/2;
  Serial.print(distanceCm);
  Serial.println("cm");

  // 감지 거리가 3m 미만일때만 소리 냄
  if(distanceCm < 300) {
    // 5cm 초과이면 뚜 뚜 뚜




    }
    else {  // 5cm 이하이면 삐이이이이이이

    }
  }
}
```

리드 스위치(reed switch)

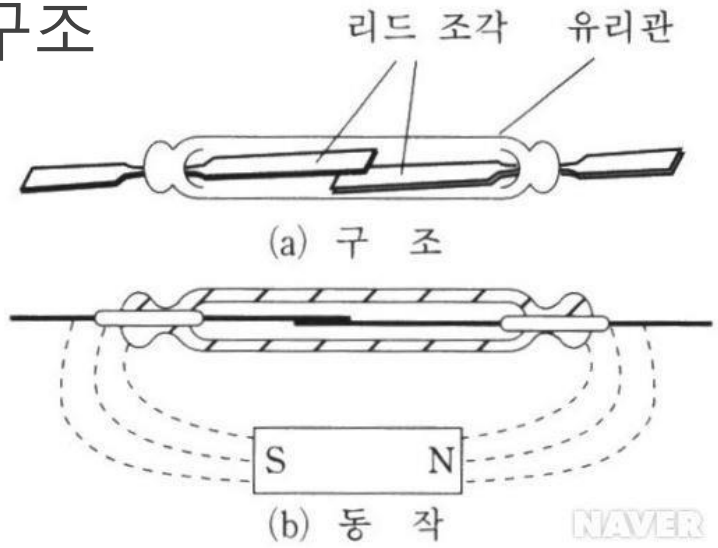
- 센서 모듈 외관



- 작동방식

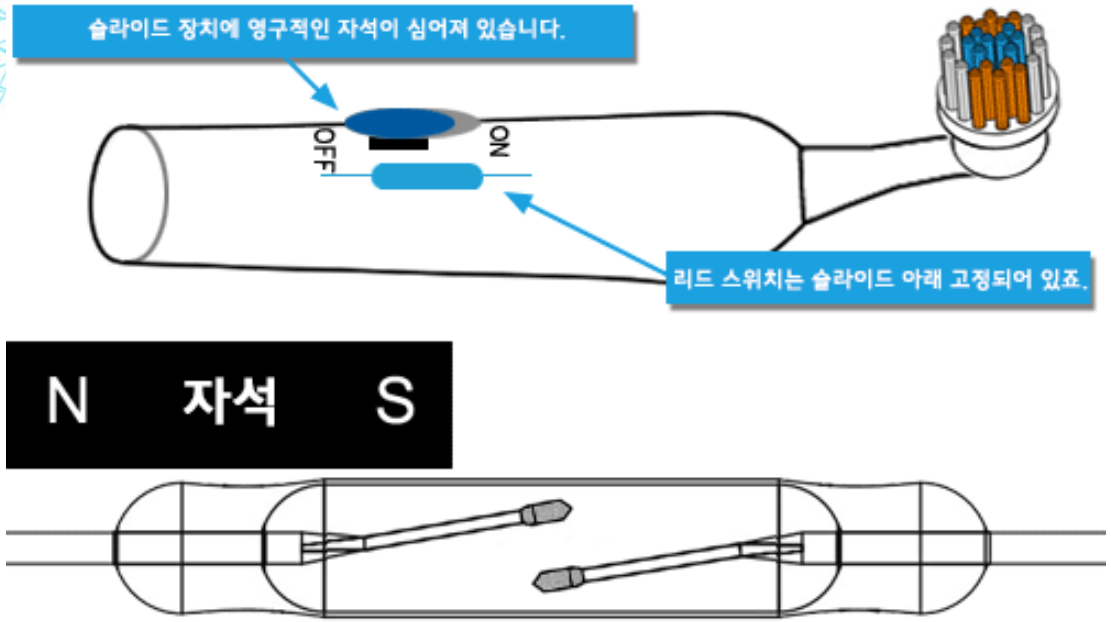


- 구조



리드 스위치(reed switch)

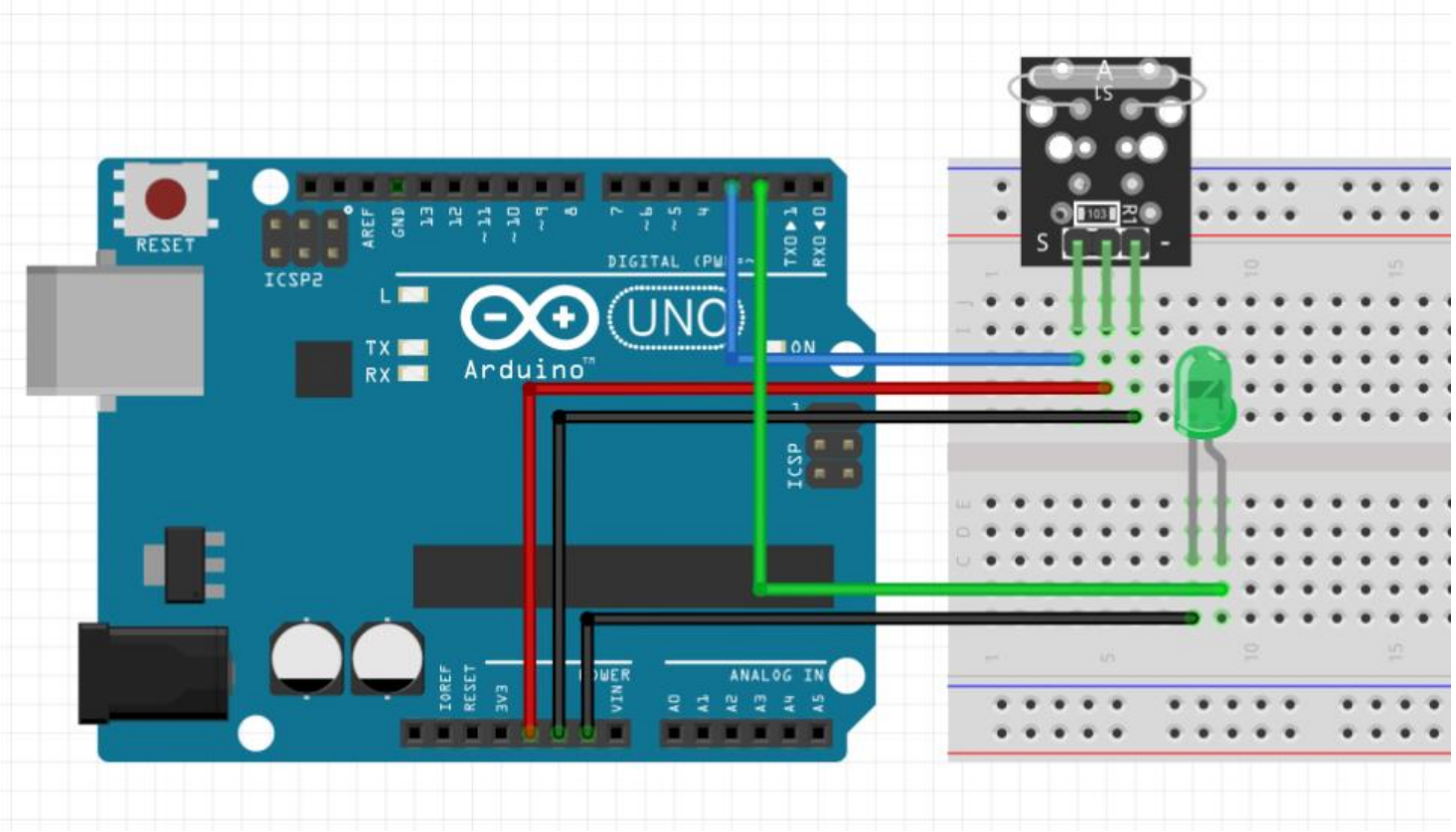
- 사용 예1



- 사용 예2

리드 스위치(reed switch)

- 회로



- 소스코드

```
// 자석이 감지되지 않으면 LED를 켠다.
#define LED_G   2
#define REEDSW  3

void setup() {
  pinMode(LED_G, OUTPUT);
  pinMode(REEDSW, INPUT);
}

void loop() {
  bool reed = digitalRead(REEDSW);
  digitalWrite(LED_G, reed);
}
```

리드 스위치(reed switch)

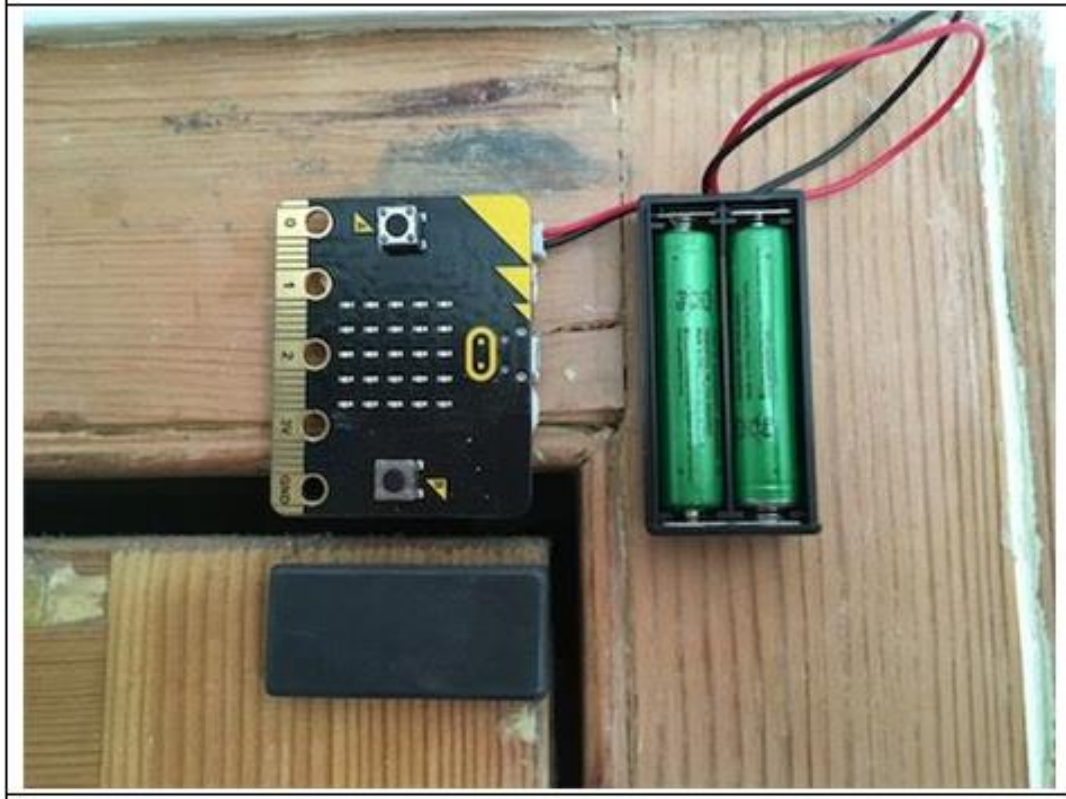
- 문 열림 감지 경보기
  - 자기장이 감지 되지 않으면 요란한 경보음이 울리는 문열림 감지 경보기를 만들어 보자.

- 소스코드

```
// 자석이 감지되지 않으면 LED를 켠다.
#define REEDSW  3
#define BUZZER  8

void setup() {
  pinMode(BUZZER, OUTPUT);
  pinMode(REEDSW, INPUT);
}

void loop() {
  bool reed = digitalRead(REEDSW);




}
```

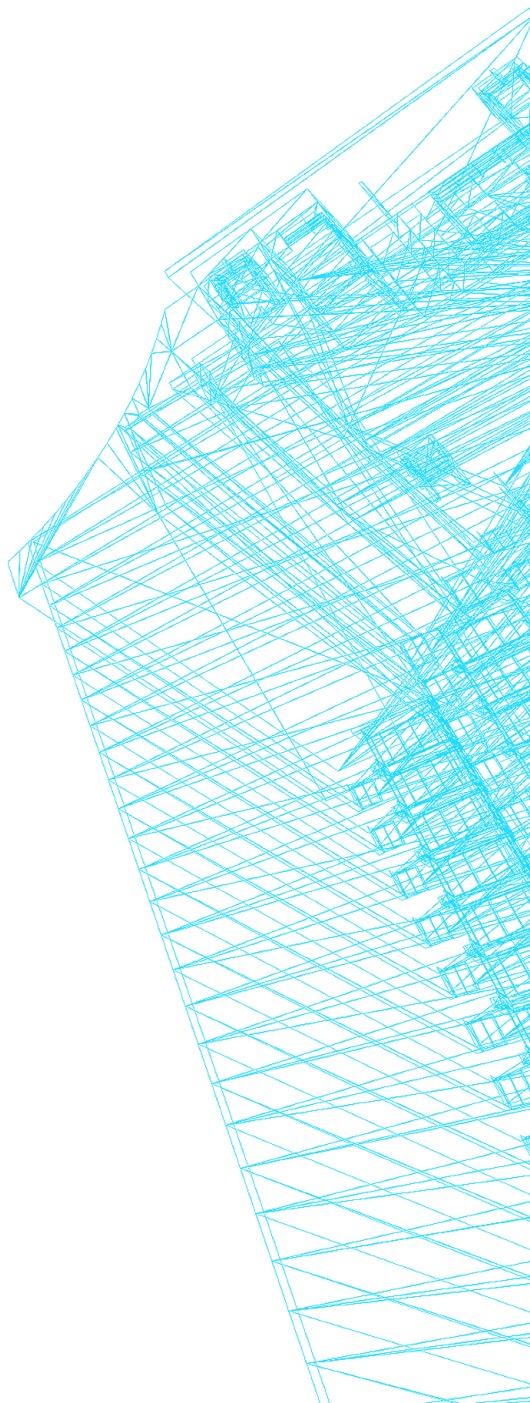
리드 스위치(reed switch)

- 문 열림 경보기의 설치 예시

(구현 예1)	(구현 예2)
	

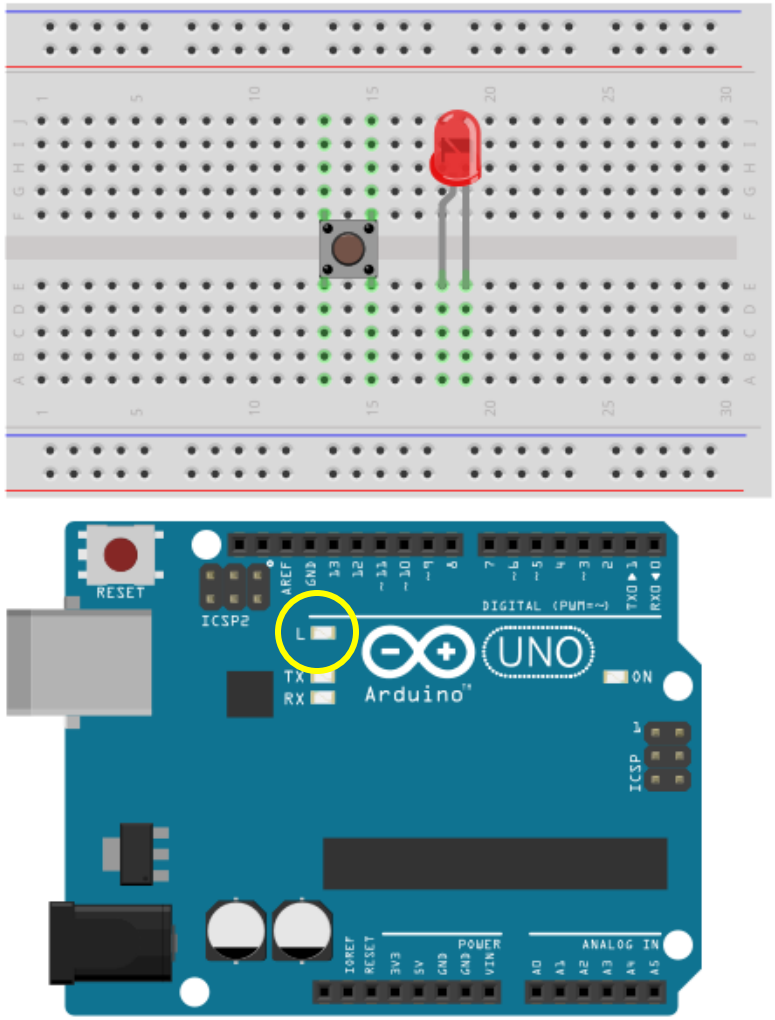
멀티태스킹 구현하기

- delay() 함수의 덫
- millis() 함수 소개
- Timer 클래스 설계
- 멀티태스킹 실험
- 타이머



delay() 함수의 덫

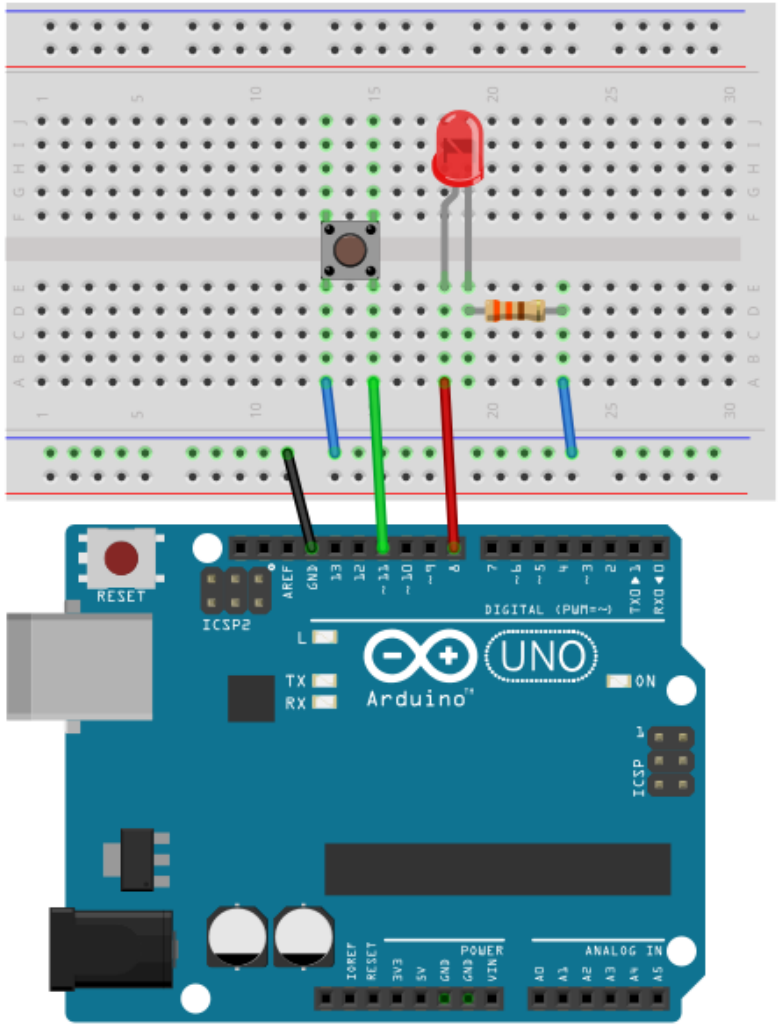
- 회로 구성



- 아래와 같은 일을 하는 회로를 구성하고, 제어 프로그램을 작성하시오.
  - 13번 핀에 연결된 내장 LED는 2초 간격으로 계속 깜박인다.
  - 11번 핀에 연결된 버튼을 누르면 8번 핀에 연결된 LED가 즉시 켜지고, 버튼에서 손을 때면 즉시 꺼진다.

delay() 함수의 덫

- 회로 구성



- 버튼
  - 11번에 연결됨
  - ACTIVE LOW로 설계
  - 내부 풀업 저항 사용
  - INPUT_PULLUP 사용해야 함.
- RED LED
  - 8번에 연결
  - ACTIVE HIGH

흔히 생각하는 구현 – 하지만 오답

- 소스코드

delay_side_effect.ino

```
#define RED  8
#define BTN 11
#define LED 13

void setup() {
  pinMode(RED, OUTPUT);
  pinMode(LED, OUTPUT);
  pinMode(BTN, INPUT_PULLUP);
  // 버튼이 눌리면 LOW
  // 버튼이 떨어지면 HIGH
}
```

```
void revertLed() {
  digitalWrite(LED, !digitalRead(LED));
  delay(2000);
  // 2초 동안 먹통이 됨.
}

void loop() {
  revertLed();
  digitalWrite(RED, !digitalRead(BTN));
}
```

millis() 함수

- 해설
  - 아두이노가 현재 프로그램의 실행을 시작한 이후 흐른 시간을 밀리초 단위로 반환한다. 이 숫자는 대략 50일 후 오버플로우(0으로 되돌아감) 될 것이다.
- 리턴
  - 프로그램이 시작된 이후 지나간 시간의 밀리초(unsigned long)
- 주의
  - millis() 함수의 리턴 값이 unsigned long 이라는 것에 주의
  - 보다 작은 타입(int 자료형 등)과 연산 시 주의 필요
- 예시

```
unsigned long currentTime = millis();
```

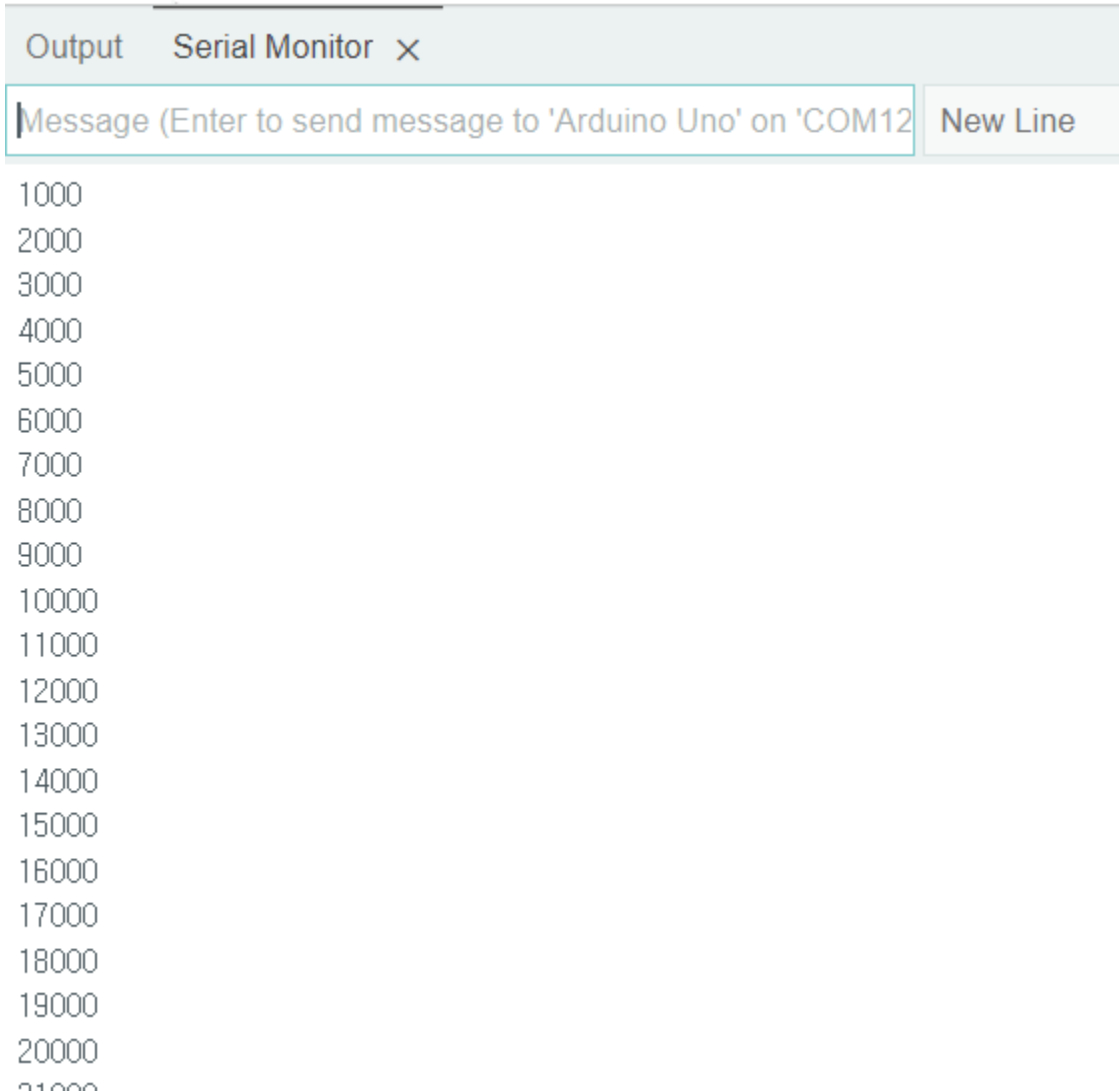
millis() 함수

- 소스코드

millis1.ino

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

int preSec = 0;

void loop() {
  unsigned long now = millis();
  int nowSec = now/1000; //초단위로 변경

  if(preSec != nowSec) {
    Serial.println(now);
    preSec = nowSec;
  }
}
```

- 실행결과



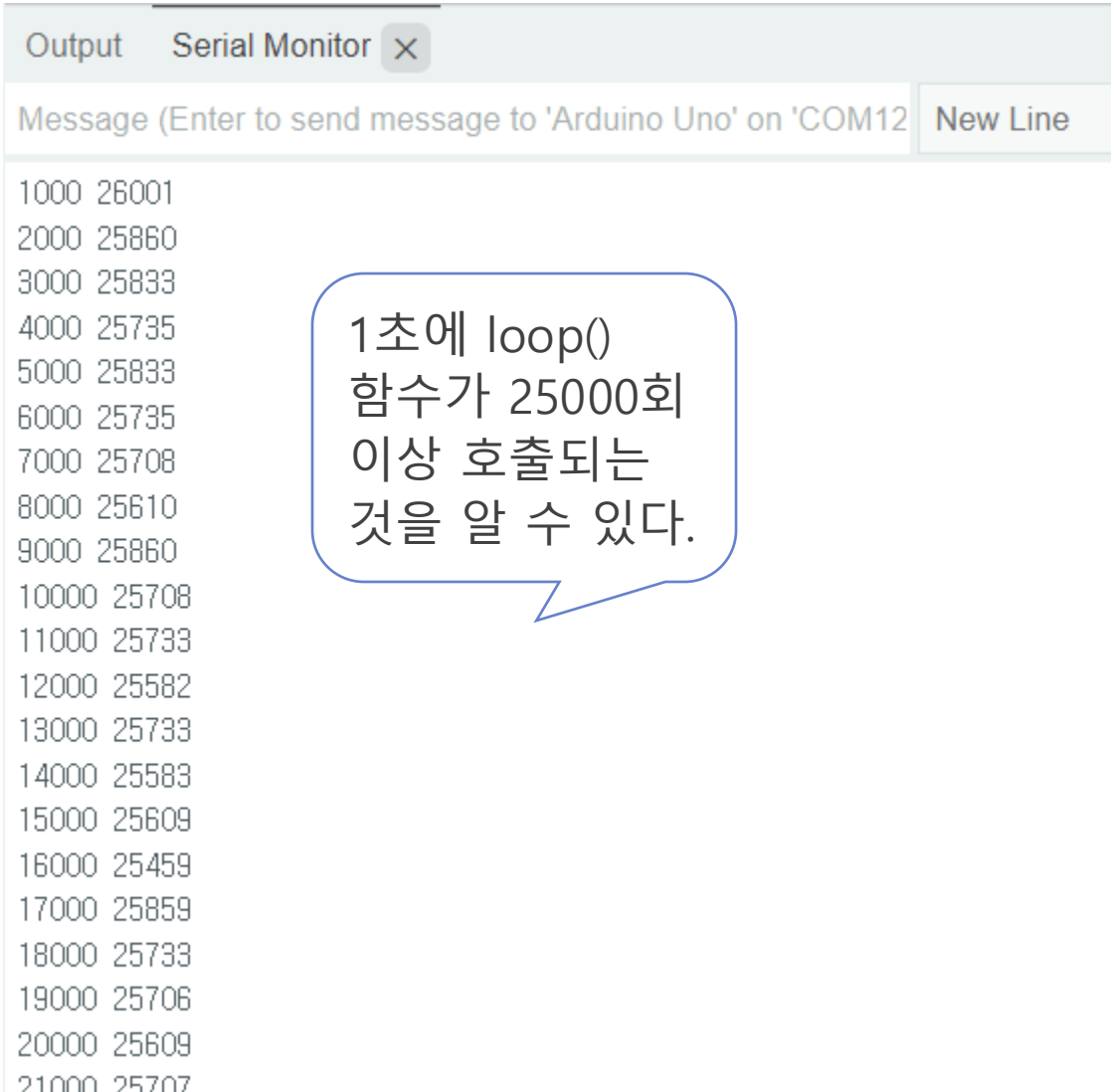
millis() 함수

- 소스코드

millis2.ino

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

int preSec = 0;
int loopCnt = 0;

void loop() {
  unsigned long now = millis();
  int nowSec = now/1000;

  if(preSec != nowSec) {
    Serial.print(now);
    preSec = nowSec;

    Serial.print(" ");
    Serial.println(loopCnt);
    loopCnt=0;
  }
  loopCnt++;
}
```

- 실행결과



1초에 loop() 함수가 25000회 이상 호출되는 것을 알 수 있다.

delay() 함수의 덫 해답

- millis() 함수 이용 흐른 시간을 측정하여 반전

delay_x.ino

```
#define RED  8
#define LED 13
#define BTN 11

void setup() {
  pinMode(RED, OUTPUT);
  pinMode(LED, OUTPUT);
  pinMode(BTN, INPUT_PULLUP);
}

void loop() {
  revertLED();
  digitalWrite(RED, !digitalRead(BTN));
}
```

```
unsigned long preTime = 0;

void revertLED() {
  unsigned long curTime = millis();
  // curTime = 현재시간

  //(현재시각 - 이전시간 >= 2초)
  if(curTime - preTime >= 2000) {  // 2초 지났니?
    digitalWrite(LED, !digitalRead(LED));
    preTime = curTime;
  }
}
```

```
void revertLed() {
  digitalWrite(LED, !digitalRead(LED));
  delay(2000);  // 2초 동안 먹통이 됨.
}
```

millis()를 이용한 방법 고찰

- 문제점
  - delay() 함수를 회피하기 위해서 매번 시간 백업 변수를 만들고 현재 시간을 측정하여 원하는 시간이 흘렀는지 감시하는 루틴의 작성이 요구됨.
  - 시간에 의존하는 작업이 1개 늘어 날 때마다,
  - 시간 백업을 위한 전역 변수를 또 만들고 시간을 감시하는 루틴을 계속 작성해야 하므로 소스코드가 복잡해짐.
- 해결책
  - 정해진 시간마다 원하는 함수를 자동으로 호출해주는 시간관리 모듈이 필요함.
  - 호출해야 하는 함수와 시간간격을 한 번 등록해 두면 시간 감시와 함수 호출의 자동화가 이루어져야 함.
  - 이 모든 것을 단순하게 만들어주는 클래스를 구현 해보자!!

MillisTimer 클래스 설계

MillisTimer.h

```
#define MAX_TIMERS  10
typedef unsigned long ulong;
typedef void (*timer_callback)(void);

typedef struct {
  bool enabled;
  timer_callback func;
  ulong interval;
  ulong preTime;
} timer;

static inline ulong elapsed() {
  return millis();
}

class MillisTimer {
  private:
    // 이전에 run() 함수가 실행된 시간
    static ulong _pre_run_ms;
    static ulong _refresh_ms;
    static int   _num_timers; // 등록된 타이머 개수
    static timer _timers[MAX_TIMERS];
```

	_timers
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

```
  public:
    static void add(timer_callback f,
                    ulong interval) {
      _timers[_num_timers].enabled = true;
      _timers[_num_timers].interval = interval;
      _timers[_num_timers].func = f;
      _timers[_num_timers].preTime = elapsed();
      _num_timers++;
    }

    static void enable(int n) {
      _timers[n].enabled = true;
    }

    static void disable(int n) {
      _timers[n].enabled = false;
    }

    static void get_refresh_ms() {
      return _refresh_ms;
    }
```

MillisTimer 클래스 설계

```
    // this function must be called inside loop()
    static void run() {
      ulong now_ms = elapsed();
      _refresh_ms = now_ms - _pre_run_ms;
      _pre_run_ms = now_ms;

      for(int i=0; i<_num_timers; i++) {
        if(_timers[i].enabled) {
          ulong diff = now_ms - _timers[i].preTime;
          if(diff >= _timers[i].interval) {
            (*_timers[i].func)();
            _timers[i].preTime +=timers[i].interval;
          }
        }
      }
    }
};

static ulong MillisTimer::_pre_run_ms;
static ulong MillisTimer::_refresh_ms;
static int   MillisTimer::_num_timers;
static timer MillisTimer::_timers[MAX_TIMERS];
```

MillisTimer 사용 예시

revert_millis_timer.ino

```
#include "MillisTimer.h"

#define RED  8
#define BTN 11
#define LED 13

void setup() {
  pinMode(RED, OUTPUT);
  pinMode(LED, OUTPUT);
  pinMode(BTN, INPUT_PULLUP);
  MillisTimer::add(revertLed, 2000);
}

void revertLed() {
  digitalWrite(LED, !digitalRead(LED));
}

void loop() {
  MillisTimer::run();
  digitalWrite(RED, !digitalRead(BTN));
}
```

3가지 작업 멀티태스킹 실험

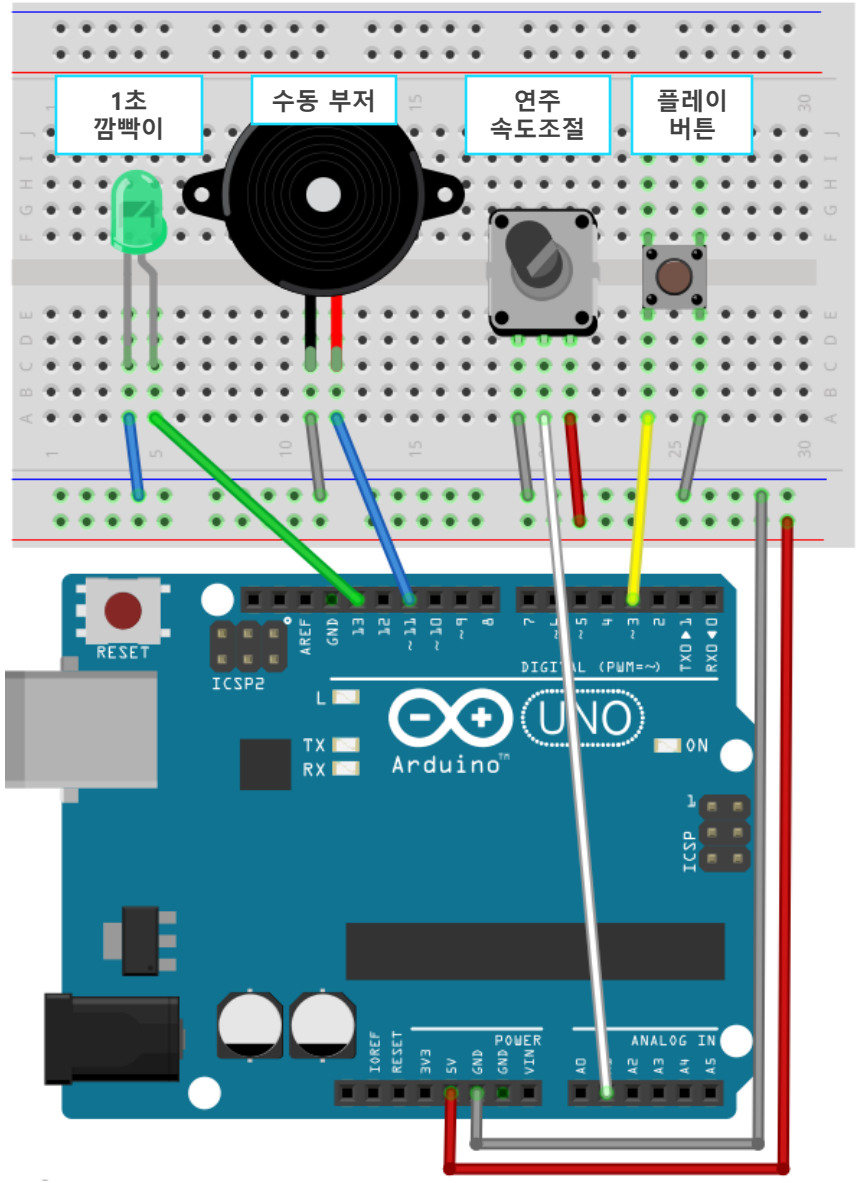
test_multitasking.ino

```ino
#include "MillisTimer.h"
#define RED  8
#define LED 13
#define BTN 11

void setup() {
  pinMode(RED, OUTPUT);
  pinMode(LED, OUTPUT);
  pinMode(BTN, INPUT_PULLUP);

  MillisTimer::add(revertLed, 500);
  MillisTimer::add(checkBtn, 1);
  MillisTimer::add(printDigitalClock, 1000);
}

void revertLed() {
  digitalWrite(LED, !digitalRead(LED));
}

void checkBtn() {
  digitalWrite(RED, !digitalRead(BTN));
}
```

```ino
void printDigitalClock() {
  int h, m, s;
  s = millis() / 1000;
  m = s / 60;
  h = s / 3600;
  s = s - m * 60;
  m = m - h * 60;
  Serial.print(h);
  printDigits(m);
  printDigits(s);
  Serial.println();
}

void printDigits(int digits) {
  Serial.print(":");
  if(digits < 10)
    Serial.print('0');
  Serial.print(digits);
}

void loop() {
  MillisTimer::run();
}
```

[MillisTimer 담당]
1) 0.5s마다 13번 LED 토글
2) 1ms마다 버튼 눌림 확인
3) 1.0s마다 디지털시계 출력

멀티태스킹 실습과제 - 주크박스



1. 13번 핀에 연결된 LED가 1초에 한 번씩 반전된다.
   - 온->오프 | 오프->온
2. 플레이 버튼을 누를 때 마다 음악 연주 상태를 변경한다.
   - 음악 연주 중에 버튼을 누르면 연주 중지
   - 연주 중지 상태에서 버튼을 누르면 연주 시작

- 2. 연주속도 조절 가변 저항으로 연주 속도를 조절
  - 왼쪽으로 돌리면 연주가 느려짐
  - 오른쪽으로 돌리면 연주가 빨라짐

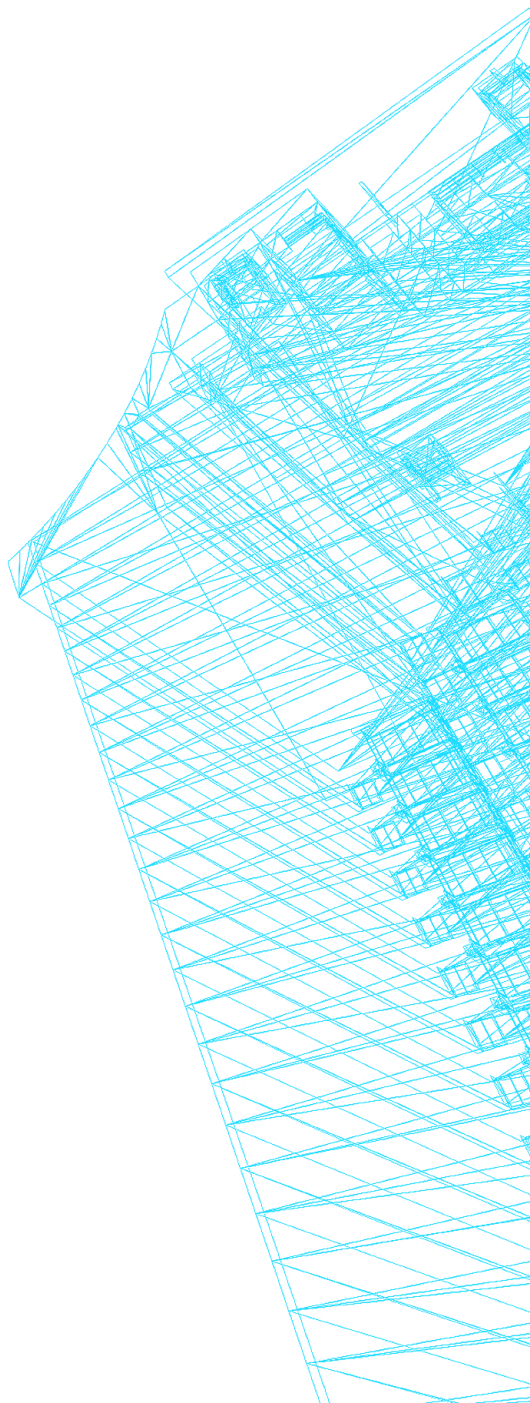
멀티태스킹 실습과제 해답 - 주크박스

jukebox.ino

```
#include "MillisTimer.h"
#include "MusicPlayer.h"

#define LED  13
#define BUZ  11
#define BTN   3
#define POT  A1

// 베토벤 환희의 송가
const char *BEETHOVEN_SCORES =
  "4 CI4 CI4 +DO4 RE4 RE4 DO4 -CI4 LA4 SO4 SO4 LA4 CI4 CI4. LA8 LA2 \
   CI4 CI4 +DO4 RE4 RE4 DO4 -CI4 LA4 SO4 SO4 LA4 CI4 LA4. SO8 SO2 \
   LA4 LA4 CI4 SO4 LA4 CI8 +DO8 -CI4 SO4 LA4 CI8 +DO8 -CI4 LA4 SO4 LA4 RE4 CI4 \
   CI4 CI4 +DO4 RE4 RE4 DO4 -CI4 LA4 SO4 SO4 LA4 CI4 LA4. SO8 SO2 \
   LA4 LA4 CI4 SO4 LA4 CI8 +DO8 -CI4 SO4 LA4 CI8 +DO8 -CI4 LA4 SO4 LA4 RE4 CI4 \
   CI4 CI4 +DO4 RE4 RE4 DO4 -CI4 LA4 SO4 SO4 LA4 CI4 LA4. SO8 SO2 ZZ2";

MusicPlayer bgmPlayer(BUZ);

void invertLed() {  // 13번 LED 1초 마다 토글
  digitalWrite(LED, !digitalRead(LED));
}

void checkPlayBtn() {  // 연주를 시작 또는 중단
  static bool btn_pre = HIGH;
  bool btn_now = digitalRead(BTN);

  if(btn_pre==HIGH && btn_now==LOW) {
    bool e = bgmPlayer.enabled();
    bgmPlayer.setEnable(!e);
  }
  btn_pre = btn_now;
}

void checkTempoPot() {  // 가변 저항값으로 템포 조절
  int pot = analogRead(POT);
  int temp = map(pot, 0, 1024, 60, 480);
  bgmPlayer.setTempo(temp);
}

void playBgm() {
  bgmPlayer.play();
}

void setup() {
  pinMode(LED, OUTPUT);
  pinMode(BUZ, OUTPUT);
  pinMode(BTN, INPUT_PULLUP);
  Serial.begin(9600);

  bgmPlayer.setScore(BEETHOVEN_SCORES);
  bgmPlayer.setTempo(120);
  bgmPlayer.setVerbose(true);
  bgmPlayer.start();

  MillisTimer::add(invertLed, 1000);
  MillisTimer::add(checkPlayBtn, 1);
  MillisTimer::add(checkTempoPot, 1);
  MillisTimer::add(playBgm, 1);
}

void loop() {
  MillisTimer::run();
}
```

LCD & OLED

- 1602 LCD
  - 1602 Chracter LCD
  - IIC interface
- 1306 SSD 128x64 OLED
  - 128x64 OLED
  - U8g2 lib

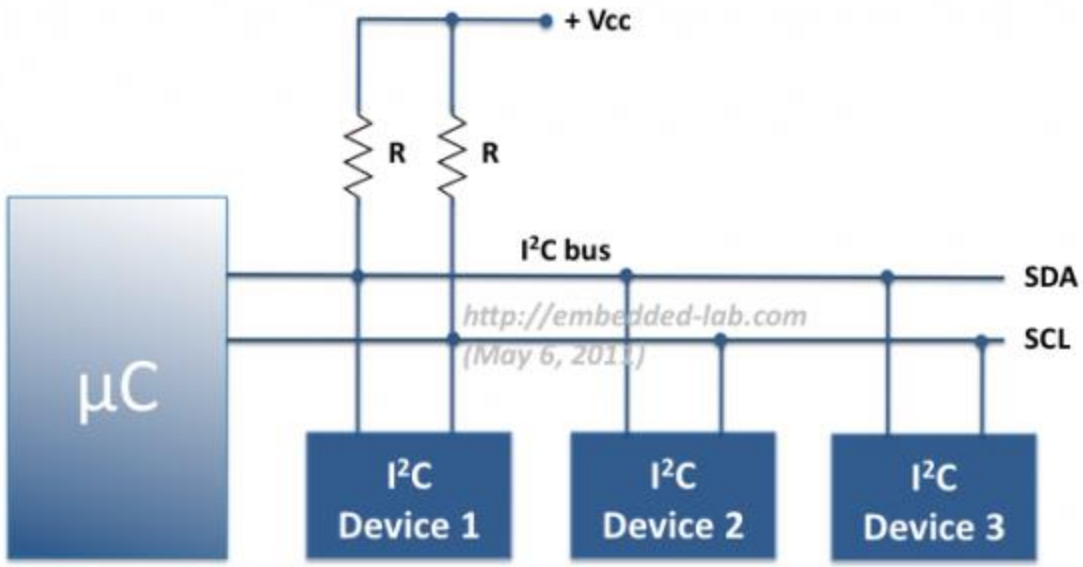


INTER-INTEGRATED CIRCUIT (I2C) COMM.

- I2C 개요와 특징
  - I2C(Inter-Integrated Circuit)는 "아이-투-씨"라고 쉽게 읽기도 하지만, 정식 발음은 "아이-스퀘어-씨(I²C)"이다. 아두이노의 칩 제조사인 'Atmel'에서는 TWI(The Two-Wire Serial Interface)라고도 한다.
  - I2C 통신은 데이터를 주고 받기 위한 선(SDA) 하나와 송수신 타이밍 동기화를 위한 클럭 선(SCL) 하나로 이루어진다.
  - 하나의 마스터(Master)와 하나 이상의 슬레이브(Slave)로 이루어지며, 슬레이브는 최대 127개까지 연결할 수 있다.
  - 데이터 전송은 bus 상태가 busy가 아닐 때 시작 가능하며, non busy 상태란 SCL, SDA 모두 High 인 경우다.
  - 동기화 통신이다. 동기화 통신 방식이란 클럭(Clock) 신호를 이용해서 데이터의 전송 타이밍을 맞추는 방식인데, 때문에 시리얼 통신처럼 통신 속도가 따로 정해지지 않아도 된다는 장점이 있다.
  - 반이중 통신이다. 따라서 보내는 동안에는 수신이 불가능하다.

INTER-INTEGRATED CIRCUIT (I2C) COMM.

- I2C Bus

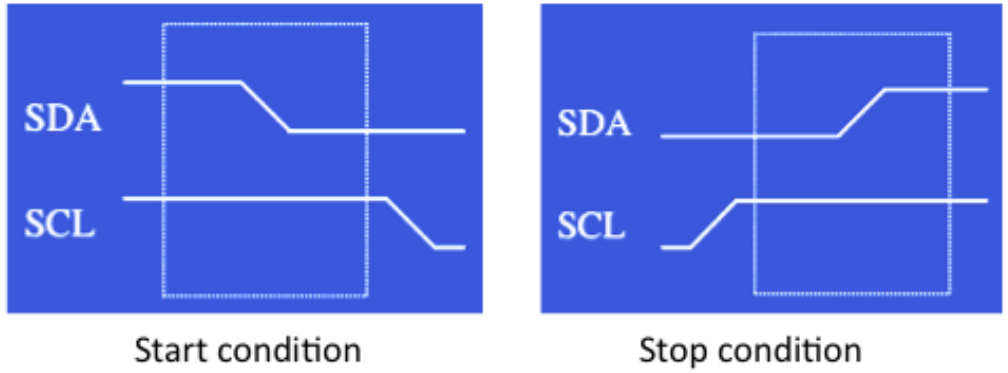


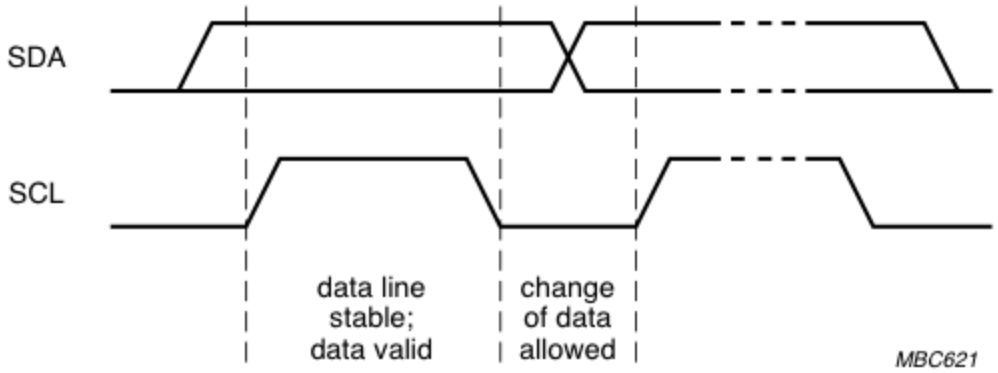
Multiple devices on common I²C bus

- SCL 및 SDA 라인은 모두 오픈 드레인 드라이버이므로 풀업 저항을 통해 플러스 전원 전압에 연결됩니다. 이는 *I2C 디바이스가 라인을 로우로 끌어내릴 수는 있지만 하이로 구동 할 수 없다는* 것을 의미합니다. 어떤 장치도 라인을 당기지 않으면 풀업 저항을 통해 하이로 플로팅됩니다. 이것이 I2C에서 풀업 저항이 중요한 이유입니다. I2C 버스의 데이터는 최대 100Kbps (표준 모드), 400Kbps (고속 모드) 또는 최대 3.4Mbps (고속 모드)로 전송할 수 있습니다.

INTER-INTEGRATED CIRCUIT (I2C) COMM.

- Data transfer





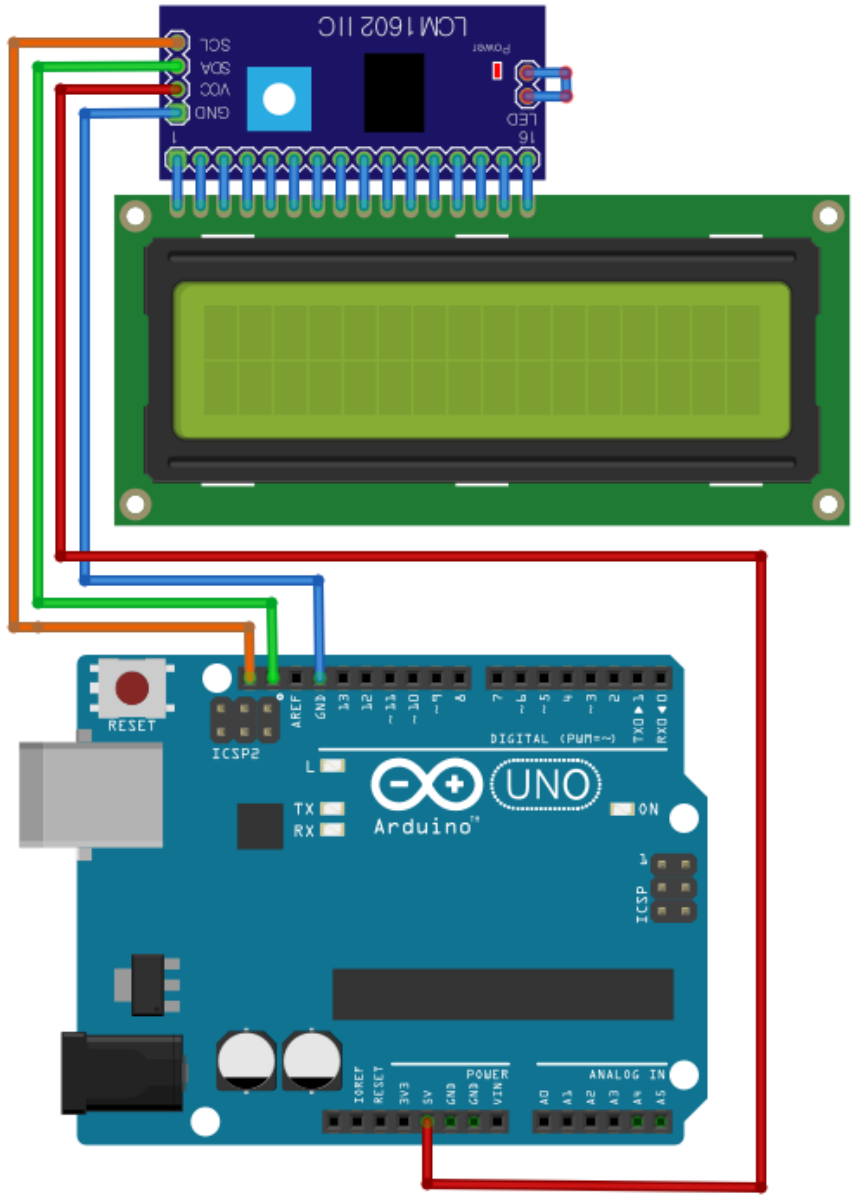
- More information
  - http://embedded-lab.com/blog/lab-14-inter-integrated-circuit-i2c-communication/

SDA 라인에 데이터를 먼저 올리고,
SCL 라인에 신호의 유효성함을 인지시킨다.

I2C ADDRESS SCANNER

```
#include <Wire.h>

void setup()  {
  Wire.begin();

  Serial.begin(9600);
  while (!Serial);        // Leonardo: wait for serial monitor
  Serial.println("\nI2C Scanner");
}

void loop()
{
  byte error, address;
  int nDevices;
  Serial.println("Scanning...");

  nDevices = 0;
  for(address = 1; address < 127; address++ )
  {
    // The i2c_scanner uses the return value of
    // the Write.endTransmisstion to see if
    // a device did acknowledge to the address.
    Wire.beginTransmission(address);
    error = Wire.endTransmission();

    if (error == 0) {
      Serial.print("I2C device found at address 0x");
      if (address<16)
        Serial.print("0");
      Serial.print(address,HEX);
      Serial.println("  !");

      nDevices++;
    }
    else if (error==4) {
      Serial.print("Unknown error at address 0x");
      if (address<16)
        Serial.print("0");
      Serial.println(address,HEX);
    }
  }

  if (nDevices == 0)
    Serial.println("No I2C devices found\n");
  else
    Serial.println("done\n");

  delay(5000);           // wait 5 seconds for next scan
}
```

LiquidCrystalDisplay 1602




```
#include <Wire.h>                       // I2C control library
#include <LiquidCrystal_I2C.h>          // LCD library
// set the LCD address to 0x3F for a 16 chars and 2line display
LiquidCrystal_I2C lcd(0x3F,16,2);
void setup() {
  lcd.init();                   // initialize the lcd
  lcd.backlight();              // turn on backlight
  lcd.print("Hello, world!");
  delay(2000);
}

int n = 1;
void loop()  {
  lcd.setCursor (0,1);          // go to start of 2nd line
  lcd.print(n++,DEC);
  lcd.setBacklight(LOW);        // Backlight off
  delay(1000);
  lcd.setBacklight(HIGH);       // Backlight on
  delay(1000);
}
```

LiquidCrystalDisplay 1602

- LiquidCrystal 라이브러리 함수

backlight	LCD 백라이트 on
nobacklight	LCD 백라이트 off
setCursor(3,0)	커서를 0번째 라인 4번째 문자로 이동
print("xxxxx")	문자열 xxxxx 를 출력
write(char)	문자 하나를 현재 커서에 출력하고 커서를 다음 위치로 이동
lcd.clear	화면 클리어
cursor	_ 커서 보임
noCursor	_ 커서 안보임
autoscroll	넘치면 스크롤
noAutoscroll	넘쳐도 스크롤 안함.
display	LCD ON
noDisplay	LCD OFF

http://arduinoliquidcrystal.readthedocs.io/en/latest/liquidcrystal.html

```
void loop() {
  // scroll 13 positions (string length) to the left to move it offscreen left:
  for (int positionCounter = 0; positionCounter < 13; positionCounter++) {
    // scroll one position left:
    lcd.scrollDisplayLeft();
    delay(150);
  }
  // scroll 29 positions (string length + display length) to the right
  // to move it offscreen right:
  for (int positionCounter = 0; positionCounter < 29; positionCounter++) {
    // scroll one position right:
    lcd.scrollDisplayRight();
    delay(150);
  }
  // scroll 16 positions (display length + string length) to the left
  // to move it back to center:
  for (int positionCounter = 0; positionCounter < 16; positionCounter++) {
    // scroll one position left:
    lcd.scrollDisplayLeft();
    delay(150);
  }
  delay(1000);
}
```

LCD Custom Character

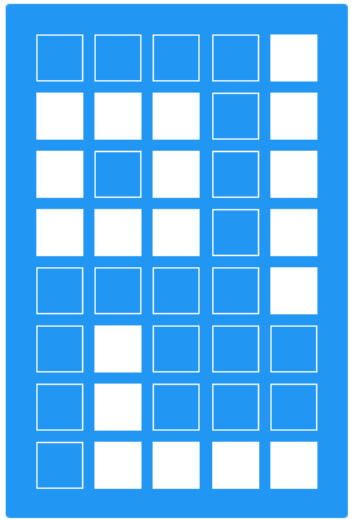
maxpromer.github.io/LCD-Character-Creator/

앱 웹프로그래밍 기타 북마크

LCD Custom Character Generator

Support character lcd and create code for Arduino.

The HD44780 is limited to 8 custom characters on screen at once:



Clear Invert

Color Green Blue

Microcontroller Arduino

Interfacing Parallel I2C

Data Type Binary Hex

Code

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

// Set the LCD address to 0x27 in PCF8574 by NXP and Set to 0x3F in PCF8574A by Ti
LiquidCrystal_I2C lcd(0x3F, 16, 2);

byte customChar[] = {
  B00001,
  B11101,
  B10101,
  B11101,
  B00001,
  B01000,
  B01000,
  B01111
};

void setup() {
  lcd.begin();
  lcd.createChar(0, customChar);
  lcd.home();
  lcd.write(0);
}

void loop() { }
```

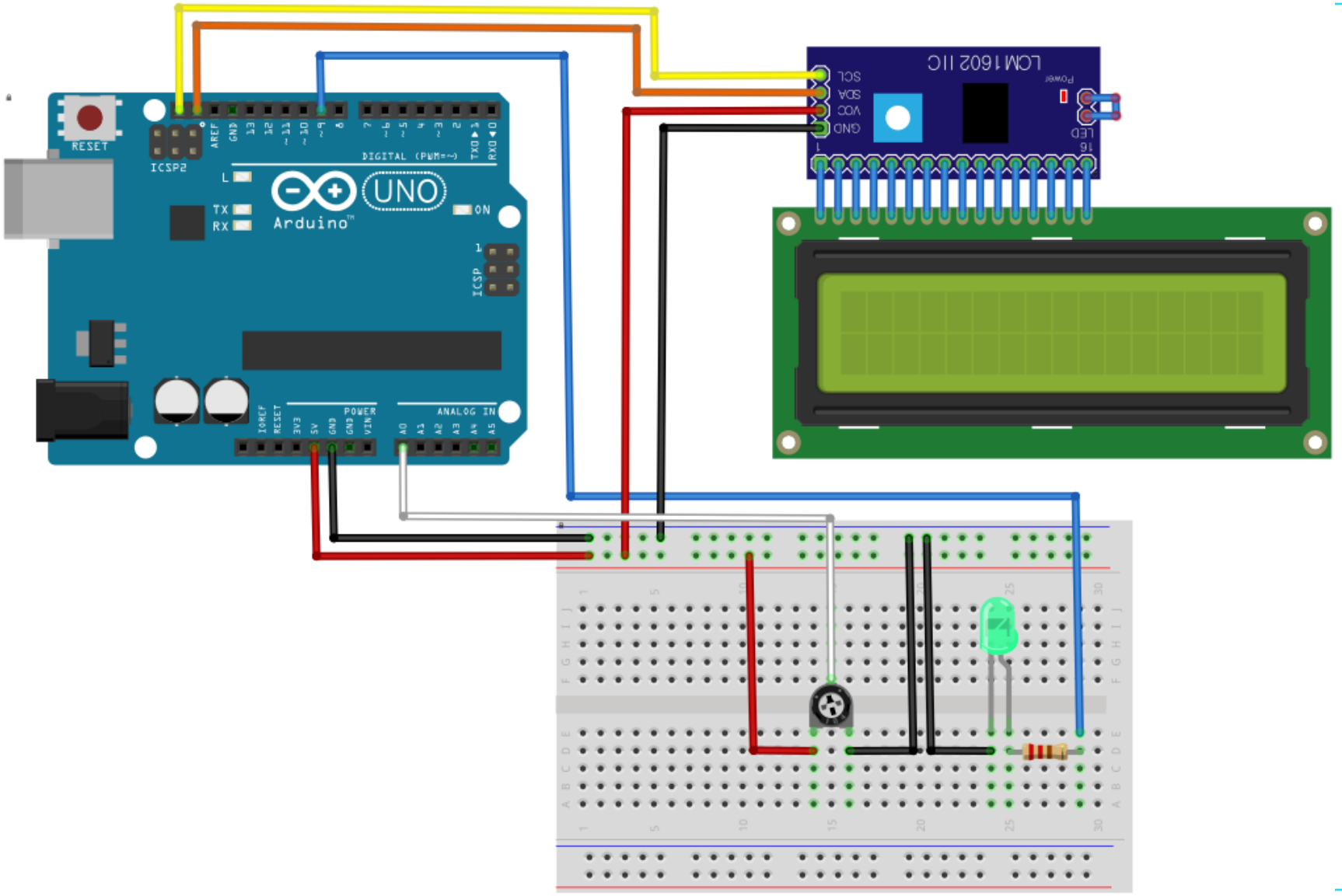
Link

- Arduino LCD Circuit
- Arduino LCD I2C Circuit
- Arduino LCD I2C library

http://maxpromer.github.io/LCD-Character-Creator/

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

// Set the LCD address to 0x27 in PCF8574 by
// NXP and Set to 0x3F in PCF8574A by Ti
LiquidCrystal_I2C lcd(0x3F, 16, 2);
byte customChar[] = {
  B00001,
  B11101,
  B10101,
  B11101,
  B00001,
  B01000,
  B01000,
  B01111
};

void setup() {
  lcd.init();
  lcd.createChar(0, customChar);
  lcd.home();
  lcd.write(0);
}

void loop() { }
```

LiquidCrystalDisplay 1602



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

LiquidCrystal_I2C lcd(0x3F, 16, 2);   // adress and format LCD

// Analog input pin that the potentiometer is attached to
const int analogInPin = A0;
const int analogOutPin = 9;      // Analog output pin
int sensorValue = 0;             // value read from the pot
int outputValue = 0;             // value output to the PWM (analog out)

void setup() {
  lcd.init();                    // init LCD
  lcd.backlight();
  lcd.home();
  lcd.setCursor(5, 0);           // place static text
  lcd.print("ADC = ");
  lcd.setCursor(15, 0);
  lcd.print("V");
  lcd.setCursor(0, 1);
  lcd.print("LED : ");
  lcd.setCursor(12, 1);
  lcd.print("%");
  Serial.begin(9600);            // optional serial monitor
}
```

```
void loop() {
  int sensorValue = analogRead(analogInPin);          // read analog A0 en store in sensorValue
  float voltValue = 5.0 / 1024.0 * sensorValue;       // calculate volt at analog A0 and store in voltValue
  float percentValue = sensorValue / 1024.0 * 100;    // calculate percentage analog A0 and store in percentValue
  outputValue = map(sensorValue, 0, 1023, 0, 255);    // map analog input to analog output and store in outputValue
  analogWrite(analogOutPin, outputValue);             // write outputValue to analogOutPin

  lcd.setCursor(0, 0);                    // set cursor at deserid position
  leadingZeros(sensorValue, 4);           // print sensorvalue with number of leading-characters
  lcd.setCursor(10, 0);                   // set cursor at deserid position
  lcd.print(voltValue);                   // print voltValue
  lcd.setCursor(6, 1);                    // set cursor at deserid position
  lcd.print(percentValue);                   // print percentValue

  Serial.print("sensor = " );              // print the results to the serial monitor:
  Serial.print(sensorValue);
  Serial.print("\t output = ");
  Serial.print(outputValue);
  Serial.print("\t procent = ");
  Serial.println(percentValue);
  delay(500);                             // set delay
}

void leadingZeros( int value, int width) {          // display result with leading characters
  char valueStr[6];                       // large enough to hold an int
  itoa (value, valueStr, 10);             // itoa converts int to string
  int len = strlen(valueStr);             // calculate length of valueStr
  if (len < width) {
    len = width - len;
    while (len--)
      lcd.print(' ');                     // for leading zeros place '0' between (), for leading spaces place ' ' between ()
  }
  lcd.print(valueStr);
}
```

I2C Interface PCF8574A 모듈

- I2C 주소 변경하기

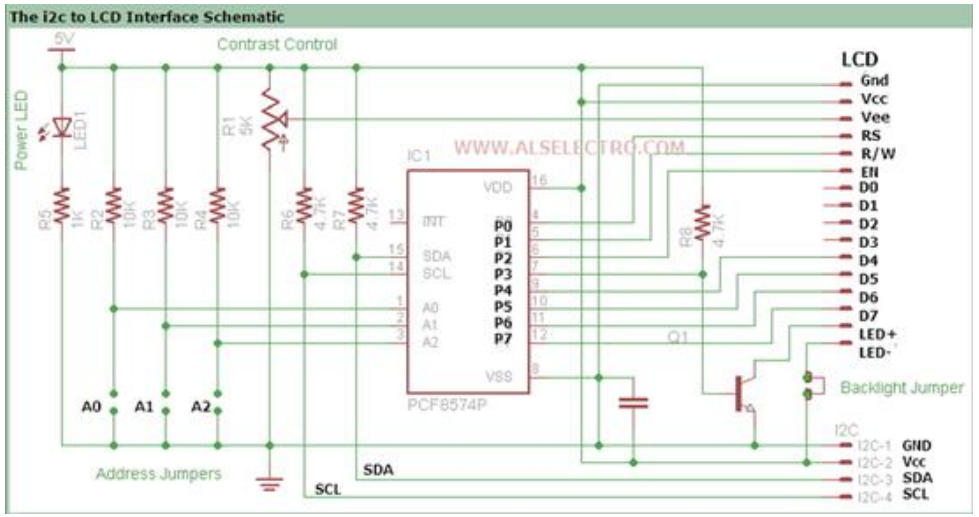
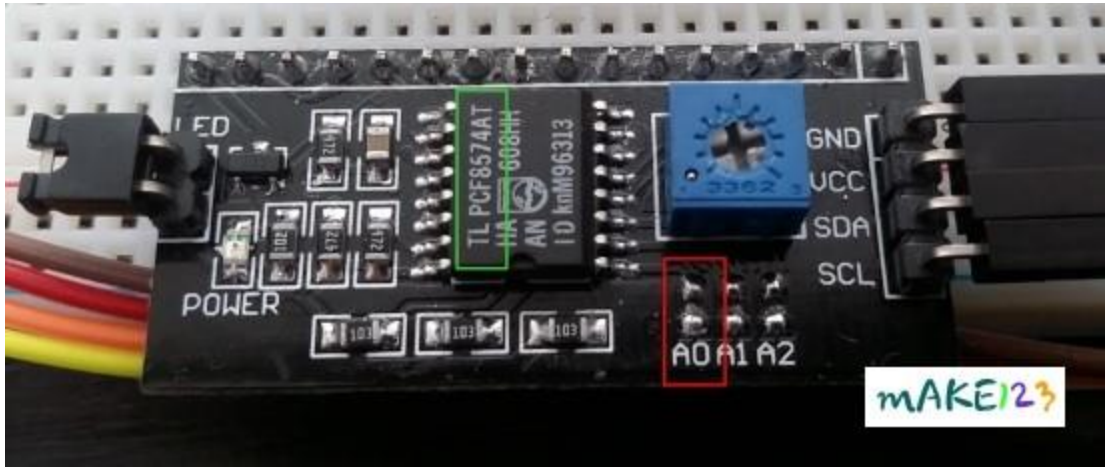
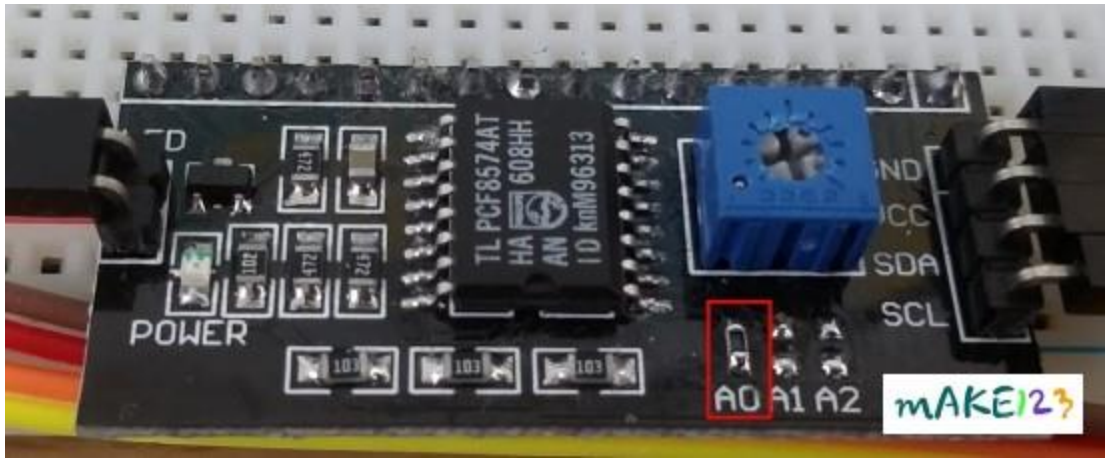


Table 5. PCF8574A address map

Pin connectivity			Address of PCF8574A								Address byte value		7-bit hexadecimal address without R/W
A2	A1	A0	A6	A5	A4	A3	A2	A1	A0	R/W	Write	Read	
V_{SS}	V_{SS}	V_{SS}	0	1	1	1	0	0	0	-	70h	71h	38h
V_{SS}	V_{SS}	V_{DD}	0	1	1	1	0	0	1	-	72h	73h	39h
V_{SS}	V_{DD}	V_{SS}	0	1	1	1	0	1	0	-	74h	75h	3Ah
V_{SS}	V_{DD}	V_{DD}	0	1	1	1	0	1	1	-	76h	77h	3Bh
V_{DD}	V_{SS}	V_{SS}	0	1	1	1	1	0	0	-	78h	79h	3Ch
V_{DD}	V_{SS}	V_{DD}	0	1	1	1	1	0	1	-	7Ah	7Bh	3Dh
V_{DD}	V_{DD}	V_{SS}	0	1	1	1	1	1	0	-	7Ch	7Dh	3Eh
V_{DD}	V_{DD}	V_{DD}	0	1	1	1	1	1	1	-	7Eh	7Fh	3Fh

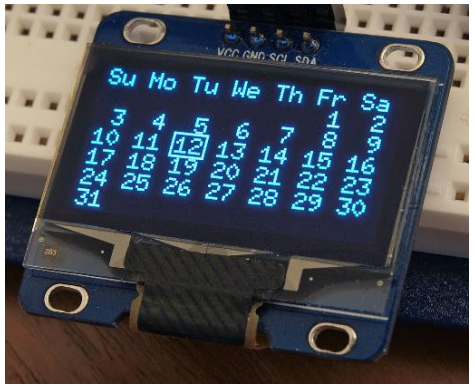
- Address jumper 납땜



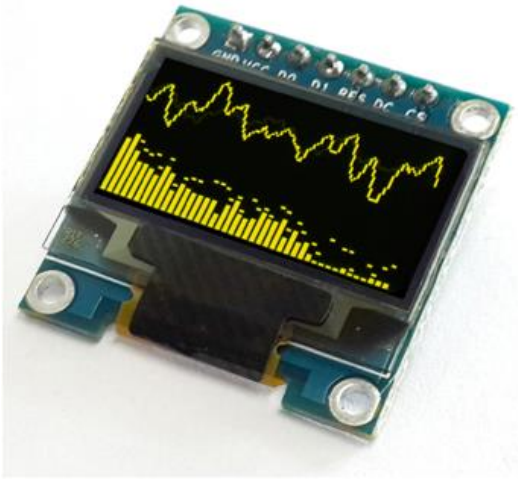


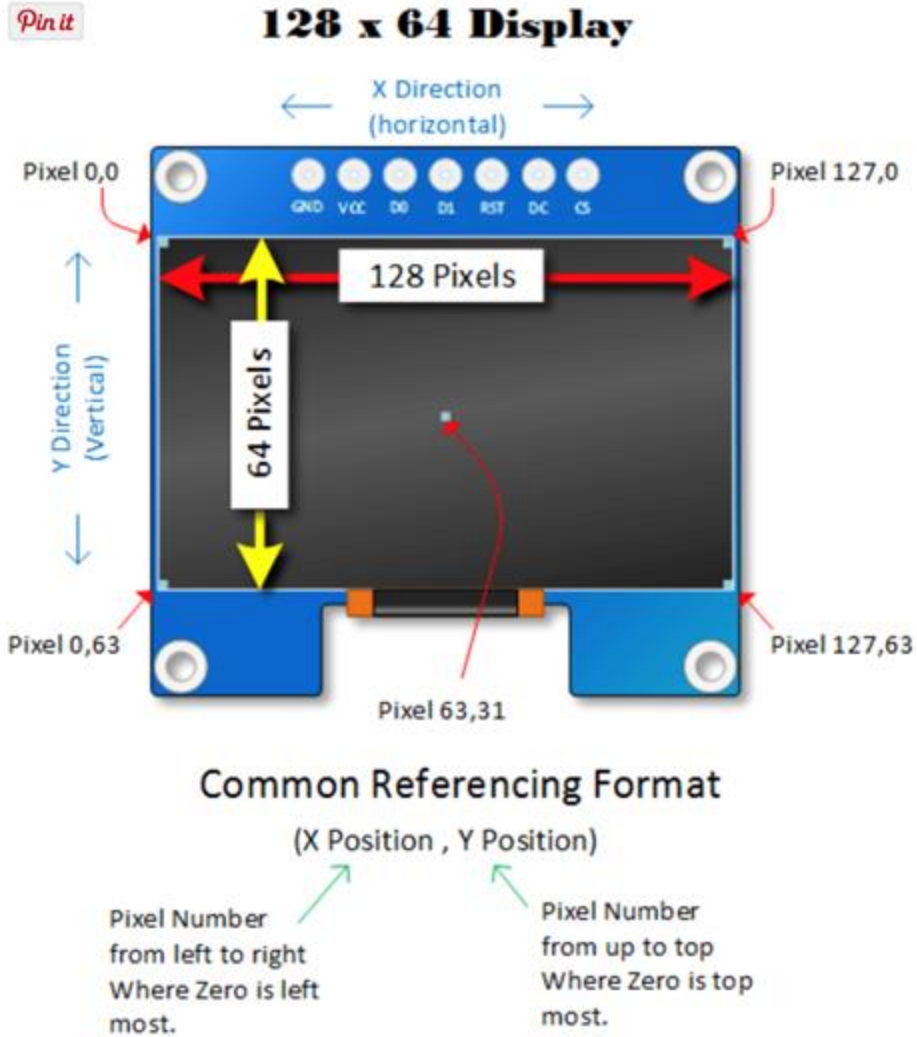
128x64 OLED

- I2C 타입



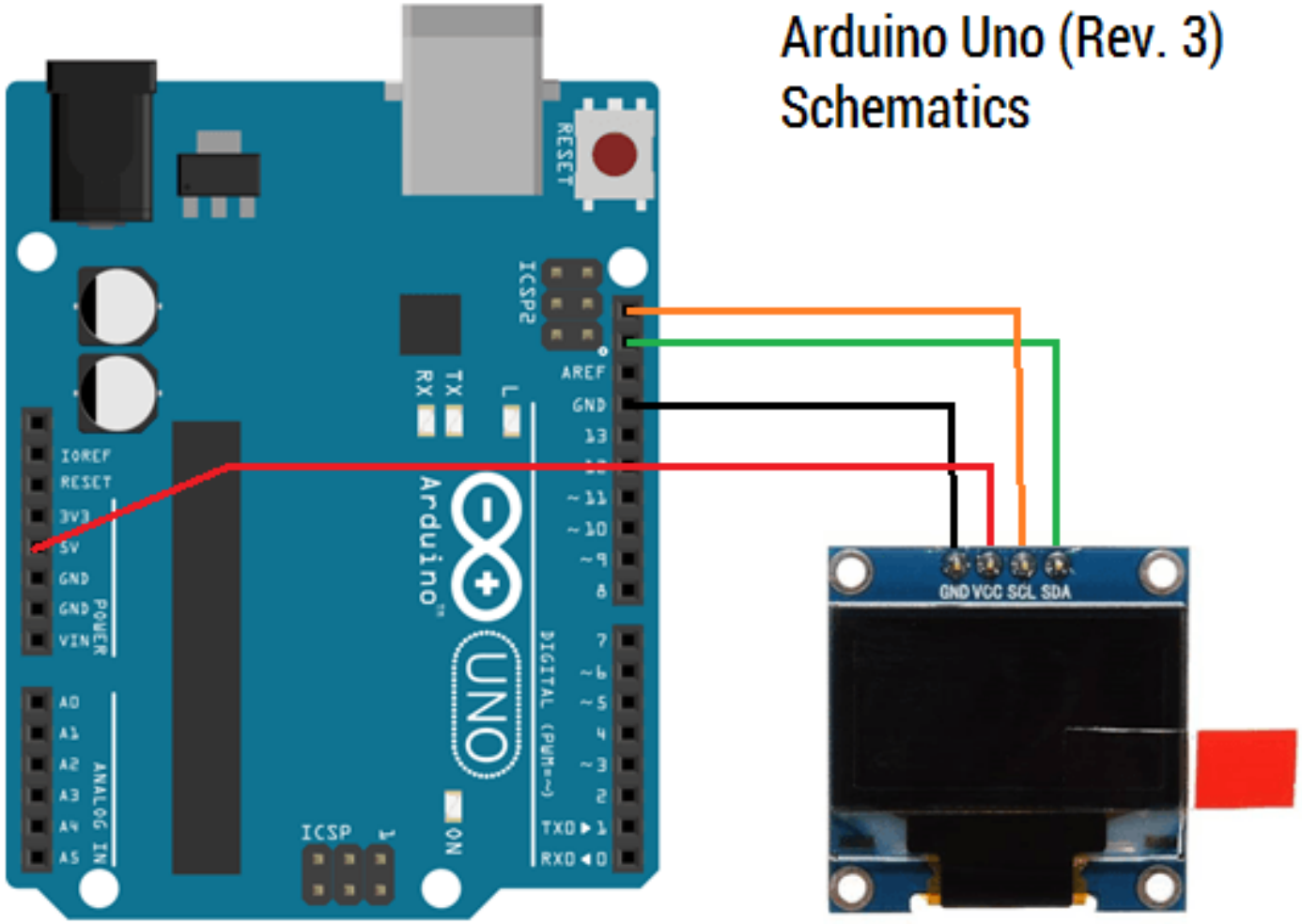
- SPI 타입





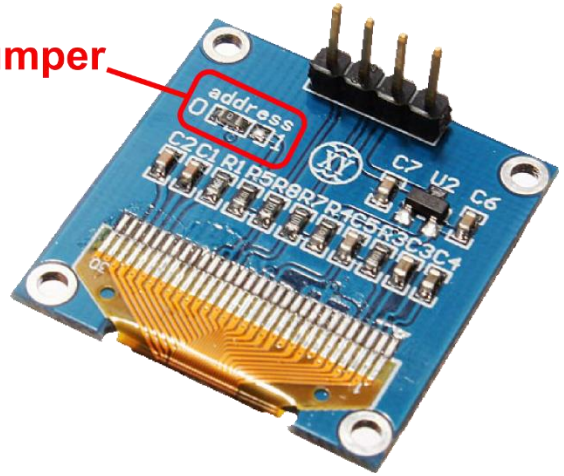
128x64 OLED

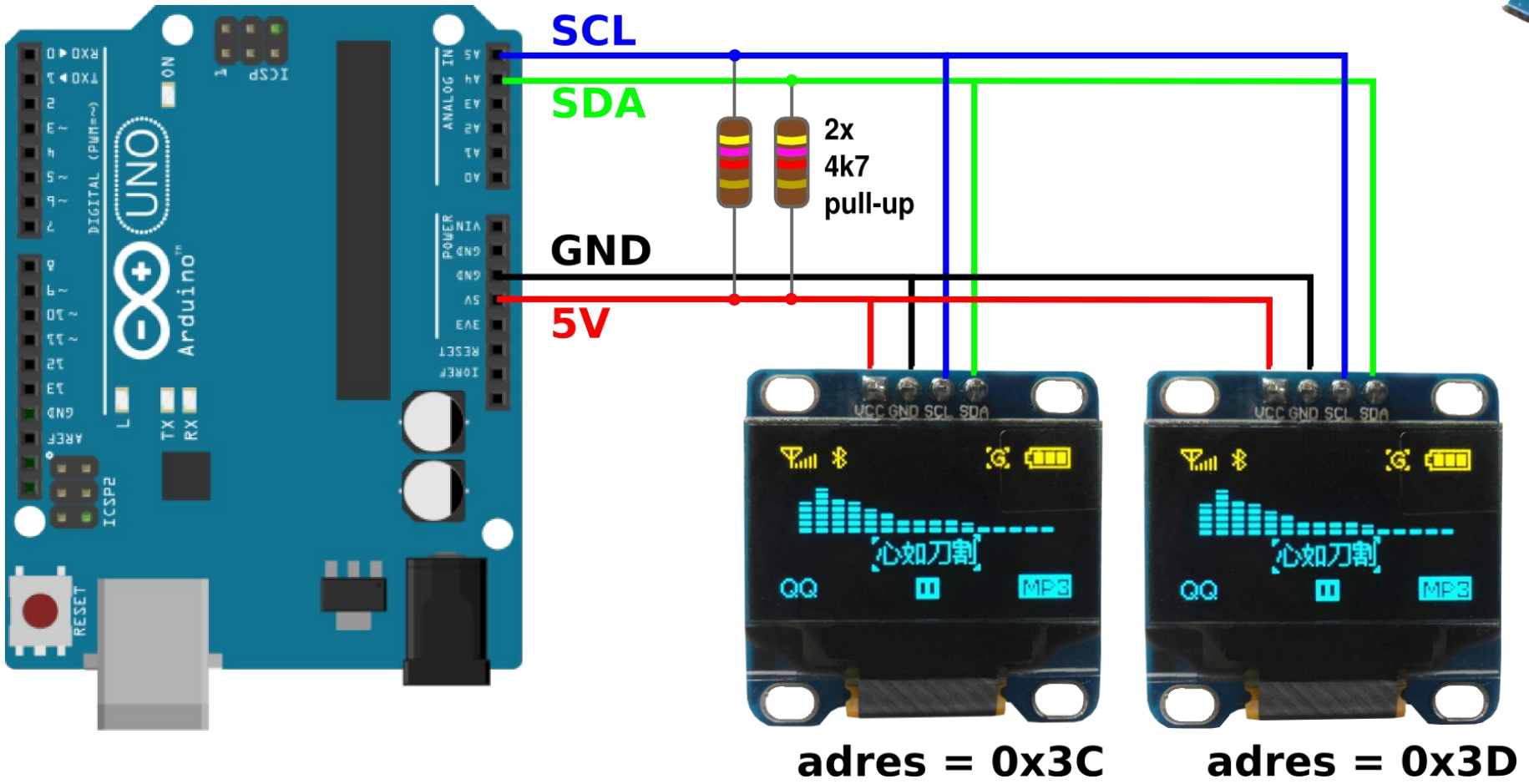
- 결선



128x64 OLED

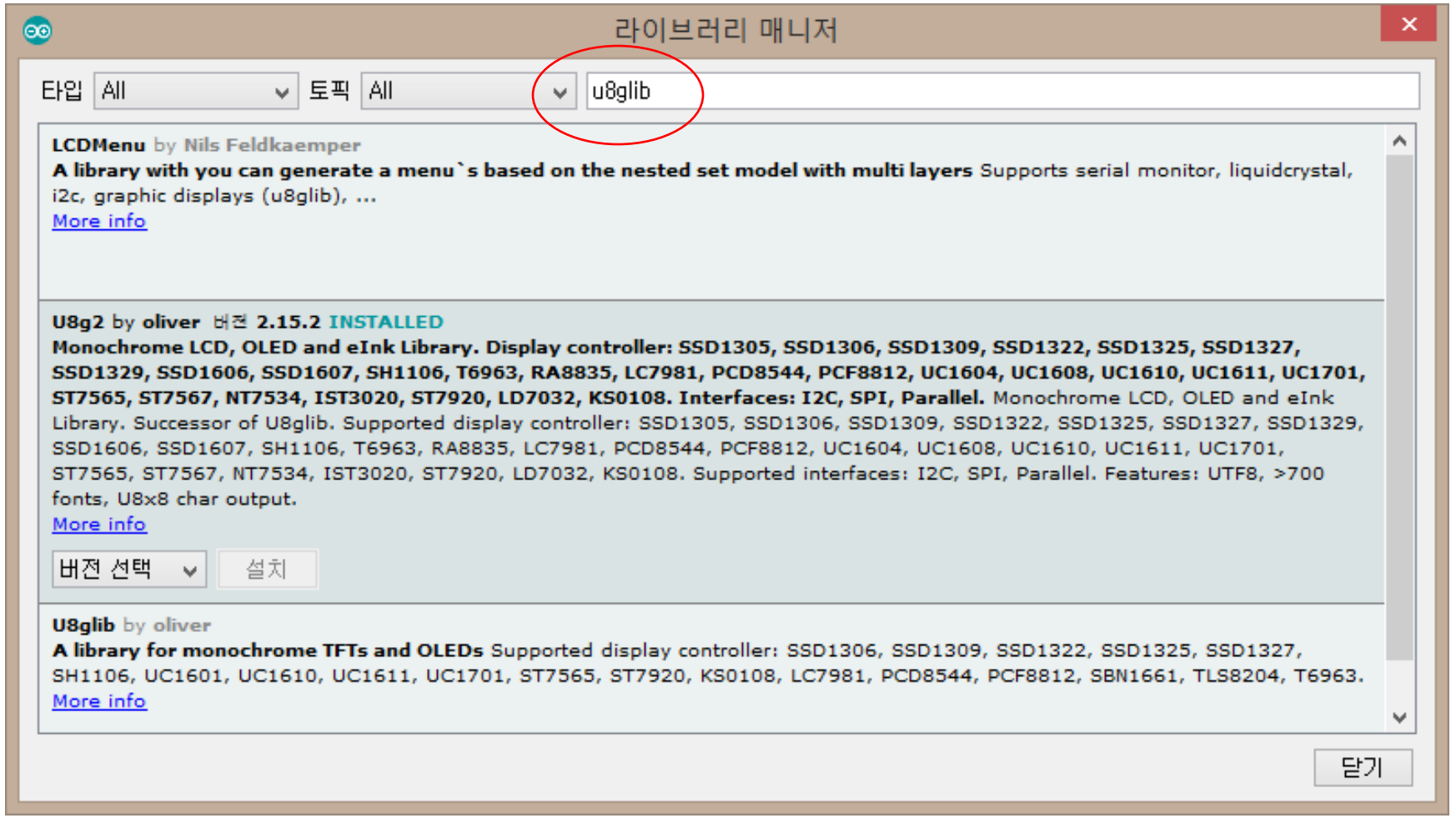
- 결선





128x64 OLED

- U8g2 라이브러리 설치



128x64 OLED

- U∞g² 라이브러리(모노크롬 디스플레이용 라이브러리, 버전 2)
  - U8g2는 컨트롤러 기반 (예 : SSD1306) 모노크롬 OLED 및 LCD를 지원.
  - 공식 사이트: https://github.com/olikraus/u8g2/wiki
  - U8g2
    - 모든 그래픽 프로시저 (라인 / 박스 / 원 그리기)가 포함됩니다.
    - 많은 글꼴을 지원합니다. (거의) 글꼴 높이에 제한이 없습니다.
    - 디스플레이를 렌더링하기 위해 마이크로 컨트롤러에 메모리가 필요합니다.
  - U8x8
    - 텍스트 출력 전용 (문자) 장치.
    - 문자 당 고정 크기로 허용되는 글꼴 만 (8x8 픽셀).
    - 디스플레이에 직접 씁니다. 마이크로 컨트롤러에 버퍼가 필요하지 않습니다.

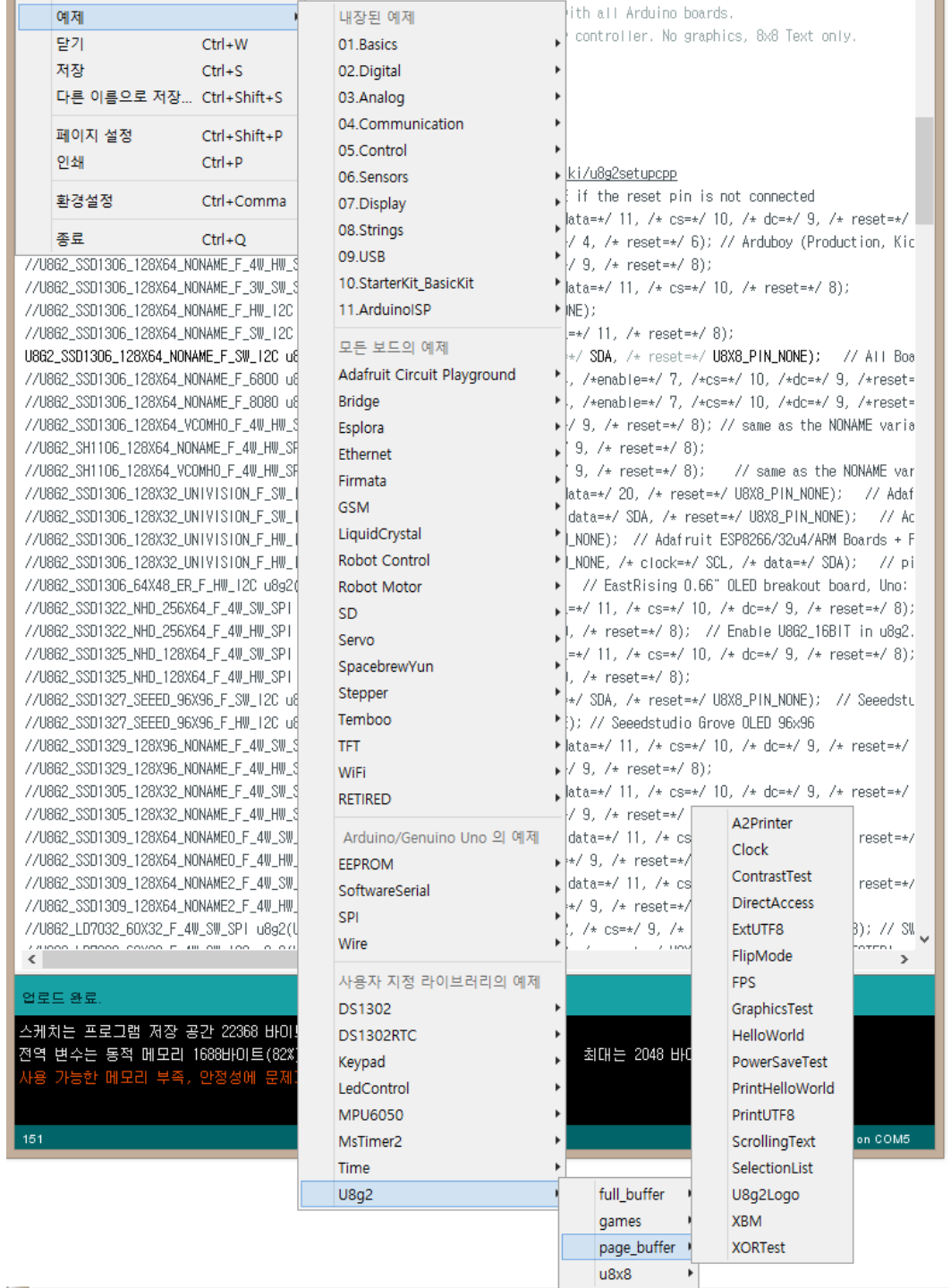
128x64 OLED

```
#include <U8g2lib.h>

U8G2_SSD1306_128X64_NONAME_F_HW_I2C u8g2(U8G2_R0,
U8X8_PIN_NONE);

void setup() {
  u8g2.begin();
  u8g2_prepare();
}

void u8g2_prepare(void) {
  u8g2.setFont(u8g2_font_6x10_tf);
  u8g2.setFontRefHeightExtendedText();
  u8g2.setDrawColor(1);
  u8g2.setFontPosTop();
  u8g2.setFontDirection(0);
}

int a = 0;
```

```
void loop() {
  u8g2.clearBuffer();
  u8g2.setFontDirection(0);
  u8g2.drawStr(30+a,31, " 0");
  u8g2.setFontDirection(1);
  u8g2.drawStr(30,31+a, " 90");
  u8g2.setFontDirection(2);
  u8g2.drawStr(30-a,31, " 180");
  u8g2.setFontDirection(3);
  u8g2.drawStr(30,31-a, " 270");
  u8g2.sendBuffer();
  delay(500);

  u8g2.clearBuffer();
  u8g2.setFontDirection(0);
  u8g2.drawStr( 0, 0, "drawBox");
  u8g2.drawBox(5,10,20,10);
  u8g2.drawBox(10+a,15,30,7);
  u8g2.drawStr( 0, 30, "drawFrame");
  u8g2.drawFrame(5,10+30,20,10);
  u8g2.drawFrame(10+a,15+30,30,7);
  u8g2.sendBuffer();
  delay(500);
  a++;
}
```

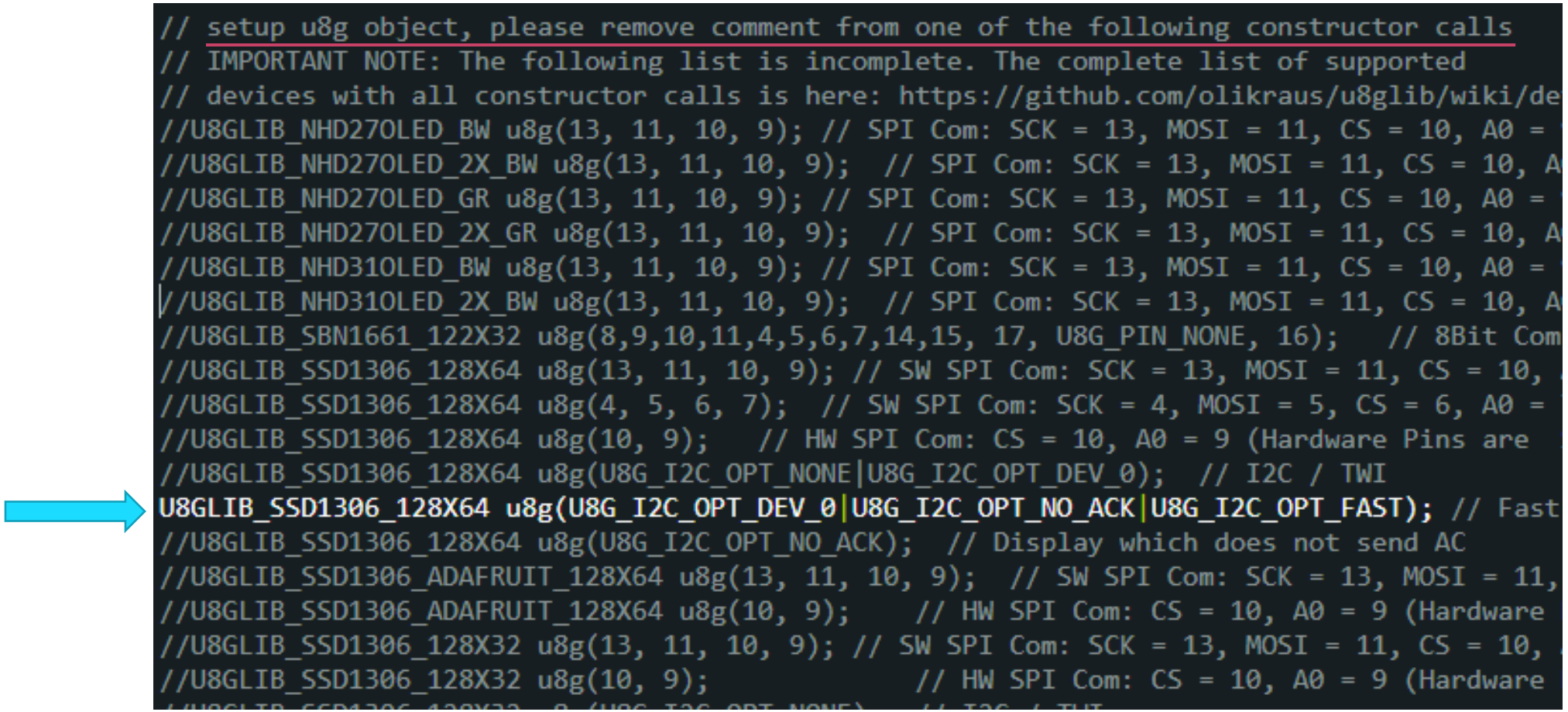
128x64 OLED

- 예제 살펴보기
  - U8g2Logo
  - GraphicsTest



128x64 OLED

- U8g2 예제 살펴보기
  - 아두이노 스케치 – 파일 – 예제 – U8g2 – full_buffer – GraphicsTest



```
// setup u8g object, please remove comment from one of the following constructor calls
// IMPORTANT NOTE: The following list is incomplete. The complete list of supported
// devices with all constructor calls is here: https://github.com/olikraus/u8glib/wiki/de
//U8GLIB_NHD27OLED_BW u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 =
//U8GLIB_NHD27OLED_2X_BW u8g(13, 11, 10, 9);  // SPI Com: SCK = 13, MOSI = 11, CS = 10, A
//U8GLIB_NHD27OLED_GR u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 =
//U8GLIB_NHD27OLED_2X_GR u8g(13, 11, 10, 9);  // SPI Com: SCK = 13, MOSI = 11, CS = 10, A
//U8GLIB_NHD31OLED_BW u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 =
//U8GLIB_NHD31OLED_2X_BW u8g(13, 11, 10, 9);  // SPI Com: SCK = 13, MOSI = 11, CS = 10, A
//U8GLIB_SBN1661_122X32 u8g(8,9,10,11,4,5,6,7,14,15, 17, U8G_PIN_NONE, 16);   // 8Bit Com
//U8GLIB_SSD1306_128X64 u8g(13, 11, 10, 9); // SW SPI Com: SCK = 13, MOSI = 11, CS = 10,
//U8GLIB_SSD1306_128X64 u8g(4, 5, 6, 7);  // SW SPI Com: SCK = 4, MOSI = 5, CS = 6, A0 =
//U8GLIB_SSD1306_128X64 u8g(10, 9);   // HW SPI Com: CS = 10, A0 = 9 (Hardware Pins are
//U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE|U8G_I2C_OPT_DEV_0);  // I2C / TWI
U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_DEV_0|U8G_I2C_OPT_NO_ACK|U8G_I2C_OPT_FAST); // Fast
//U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NO_ACK);  // Display which does not send AC
//U8GLIB_SSD1306_ADAFRUIT_128X64 u8g(13, 11, 10, 9);  // SW SPI Com: SCK = 13, MOSI = 11,
//U8GLIB_SSD1306_ADAFRUIT_128X64 u8g(10, 9);    // HW SPI Com: CS = 10, A0 = 9 (Hardware
//U8GLIB_SSD1306_128X32 u8g(13, 11, 10, 9); // SW SPI Com: SCK = 13, MOSI = 11, CS = 10,
//U8GLIB_SSD1306_128X32 u8g(10, 9);             // HW SPI Com: CS = 10, A0 = 9 (Hardware
```

128x64 OLED

- U∞g² 라이브러리(모노크롬 디스플레이용 라이브러리, 버전 2)
  - U8g2는 컨트롤러 기반 (예 : SSD1306) 모노크롬 OLED 및 LCD를 지원.
  - 공식 사이트: https://github.com/olikraus/u8g2/wiki
  - U8g2
    - 모든 그래픽 프로시저 (라인 / 박스 / 원 그리기)가 포함됩니다.
    - 많은 글꼴을 지원합니다. (거의) 글꼴 높이에 제한이 없습니다.
    - 디스플레이를 렌더링하기 위해 마이크로 컨트롤러에 메모리가 필요합니다.
  - U8x8
    - 텍스트 출력 전용 (문자) 장치.
    - 문자 당 고정 크기로 허용되는 글꼴 만 (8x8 픽셀).
    - 디스플레이에 직접 씁니다. 마이크로 컨트롤러에 버퍼가 필요하지 않습니다.

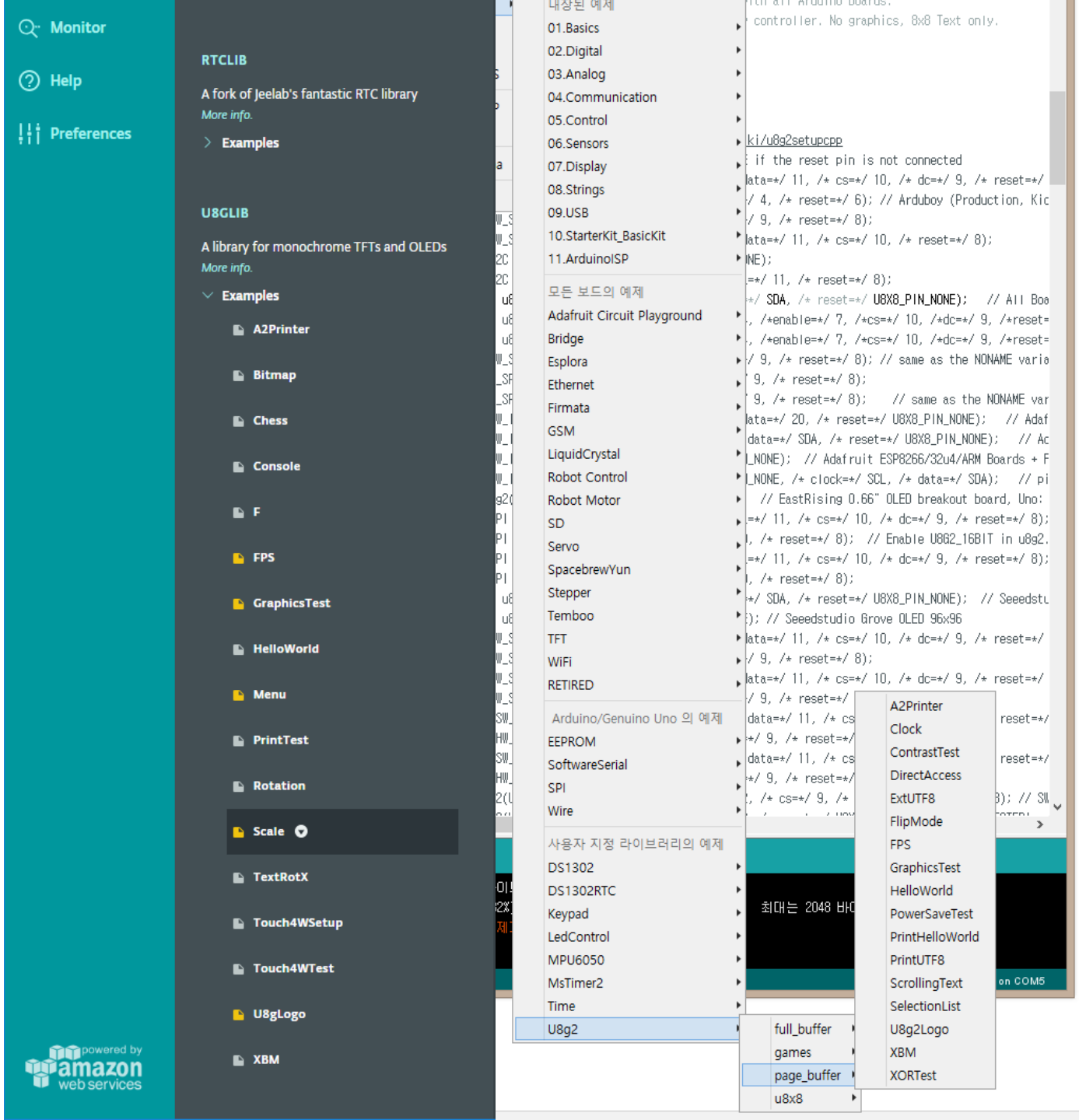
128x64 OLED

```
#include <U8g2lib.h>

U8G2_SSD1306_128X64_NONAME_F_HW_I2C u8g2(U8G2_R0,
U8X8_PIN_NONE);

void setup() {
  u8g2.begin();
  u8g2_prepare();
}

void u8g2_prepare(void) {
  u8g2.setFont(u8g2_font_6x10_tf);
  u8g2.setFontRefHeightExtendedText();
  u8g2.setDrawColor(1);
  u8g2.setFontPosTop();
  u8g2.setFontDirection(0);
}

int a = 0;
```

```
void loop() {
  u8g2.clearBuffer();
  u8g2.setFontDirection(0);
  u8g2.drawStr(30+a,31, " 0");
  u8g2.setFontDirection(1);
  u8g2.drawStr(30,31+a, " 90");
  u8g2.setFontDirection(2);
  u8g2.drawStr(30-a,31, " 180");
  u8g2.setFontDirection(3);
  u8g2.drawStr(30,31-a, " 270");
  u8g2.sendBuffer();
  delay(500);

  u8g2.clearBuffer();
  u8g2.setFontDirection(0);
  u8g2.drawStr( 0, 0, "drawBox");
  u8g2.drawBox(5,10,20,10);
  u8g2.drawBox(10+a,15,30,7);
  u8g2.drawStr( 0, 30, "drawFrame");
  u8g2.drawFrame(5,10+30,20,10);
  u8g2.drawFrame(10+a,15+30,30,7);
  u8g2.sendBuffer();
  delay(500);
  a++;
}
```

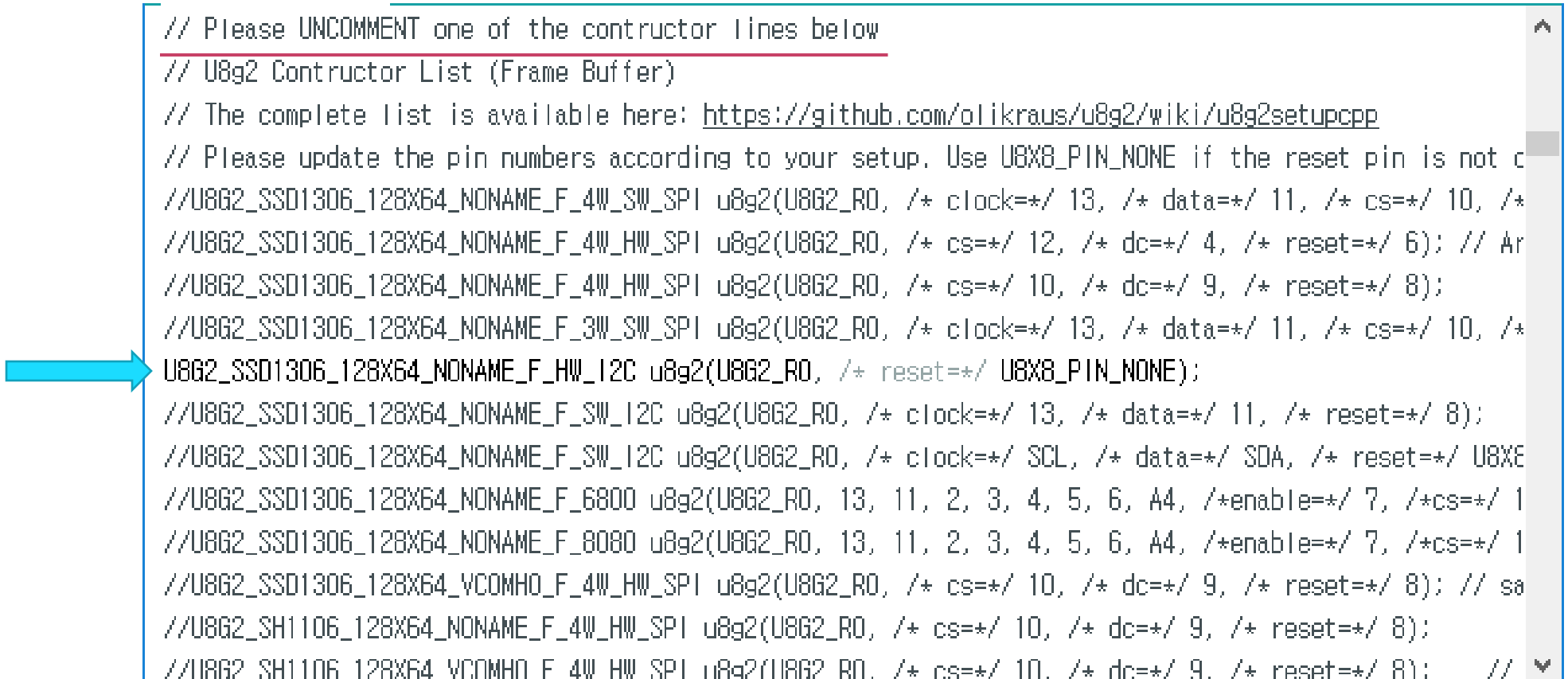
128x64 OLED

- 예제 살펴보기
  - U8g2Logo
  - GraphicsTest



128x64 OLED

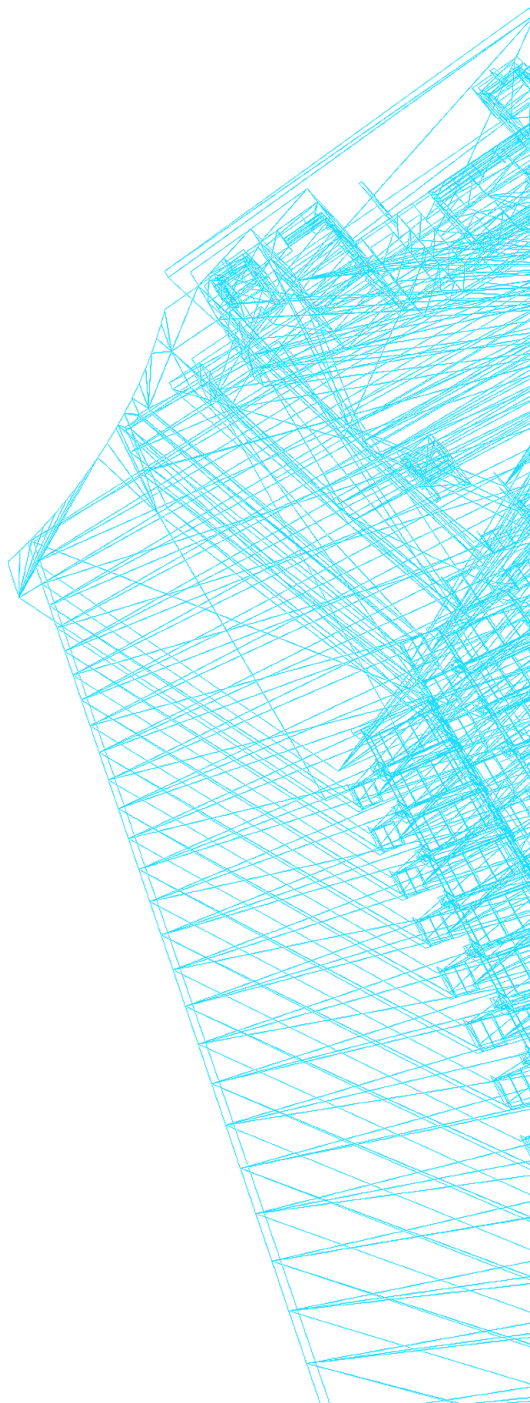
- U8g2 예제 살펴보기
  - 아두이노 스케치 – 파일 – 예제 – U8g2 – full_buffer – GraphicsTest
  - 아두이노 스케치 – 파일 – 에제 – U8g2 – U8x8 – GraphicsTest



```
// Please UNCOMMENT one of the contructor lines below
// U8g2 Contructor List (Frame Buffer)
// The complete list is available here: https://github.com/olikraus/u8g2/wiki/u8g2setupcpp
// Please update the pin numbers according to your setup. Use U8X8_PIN_NONE if the reset pin is not c
//U8G2_SSD1306_128X64_NONAME_F_4W_SW_SPI u8g2(U8G2_R0, /* clock=*/ 13, /* data=*/ 11, /* cs=*/ 10, /*
//U8G2_SSD1306_128X64_NONAME_F_4W_HW_SPI u8g2(U8G2_R0, /* cs=*/ 12, /* dc=*/ 4, /* reset=*/ 6); // Ar
//U8G2_SSD1306_128X64_NONAME_F_4W_HW_SPI u8g2(U8G2_R0, /* cs=*/ 10, /* dc=*/ 9, /* reset=*/ 8);
//U8G2_SSD1306_128X64_NONAME_F_3W_SW_SPI u8g2(U8G2_R0, /* clock=*/ 13, /* data=*/ 11, /* cs=*/ 10, /*
U8G2_SSD1306_128X64_NONAME_F_HW_I2C u8g2(U8G2_R0, /* reset=*/ U8X8_PIN_NONE);
//U8G2_SSD1306_128X64_NONAME_F_SW_I2C u8g2(U8G2_R0, /* clock=*/ 13, /* data=*/ 11, /* reset=*/ 8);
//U8G2_SSD1306_128X64_NONAME_F_SW_I2C u8g2(U8G2_R0, /* clock=*/ SCL, /* data=*/ SDA, /* reset=*/ U8X8
//U8G2_SSD1306_128X64_NONAME_F_6800 u8g2(U8G2_R0, 13, 11, 2, 3, 4, 5, 6, A4, /*enable=*/ 7, /*cs=*/ 1
//U8G2_SSD1306_128X64_NONAME_F_8080 u8g2(U8G2_R0, 13, 11, 2, 3, 4, 5, 6, A4, /*enable=*/ 7, /*cs=*/ 1
//U8G2_SSD1306_128X64_VCOMH0_F_4W_HW_SPI u8g2(U8G2_R0, /* cs=*/ 10, /* dc=*/ 9, /* reset=*/ 8); // sa
//U8G2_SH1106_128X64_NONAME_F_4W_HW_SPI u8g2(U8G2_R0, /* cs=*/ 10, /* dc=*/ 9, /* reset=*/ 8);
//U8G2_SH1106_128X64_VCOMH0_F_4W_HW_SPI u8g2(U8G2_R0, /* cs=*/ 10, /* dc=*/ 9, /* reset=*/ 8);    //
```

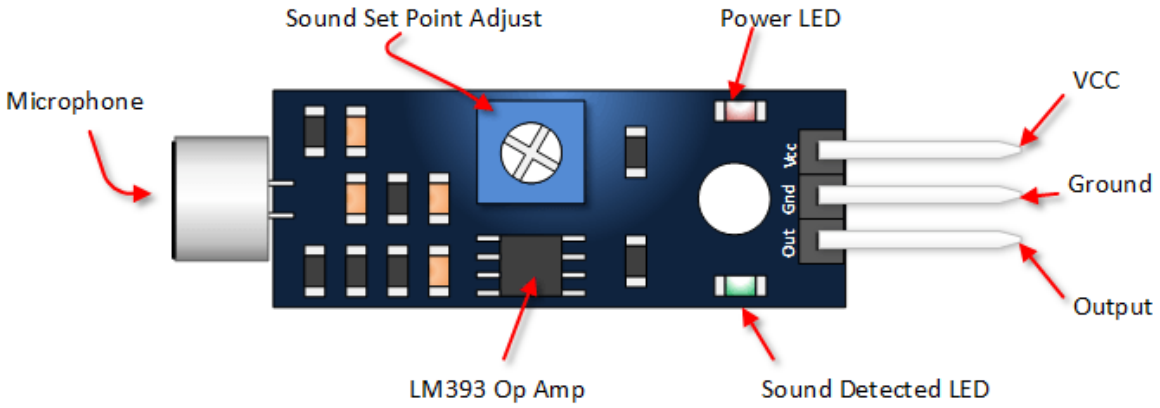
다양한 센서 모듈

- sound sensor
- joystick
- 8x8 led matrix
- DHT11 (온·습도 센서)
- 4x4 matrix keypad



Sound sensor(detector)

- 외관

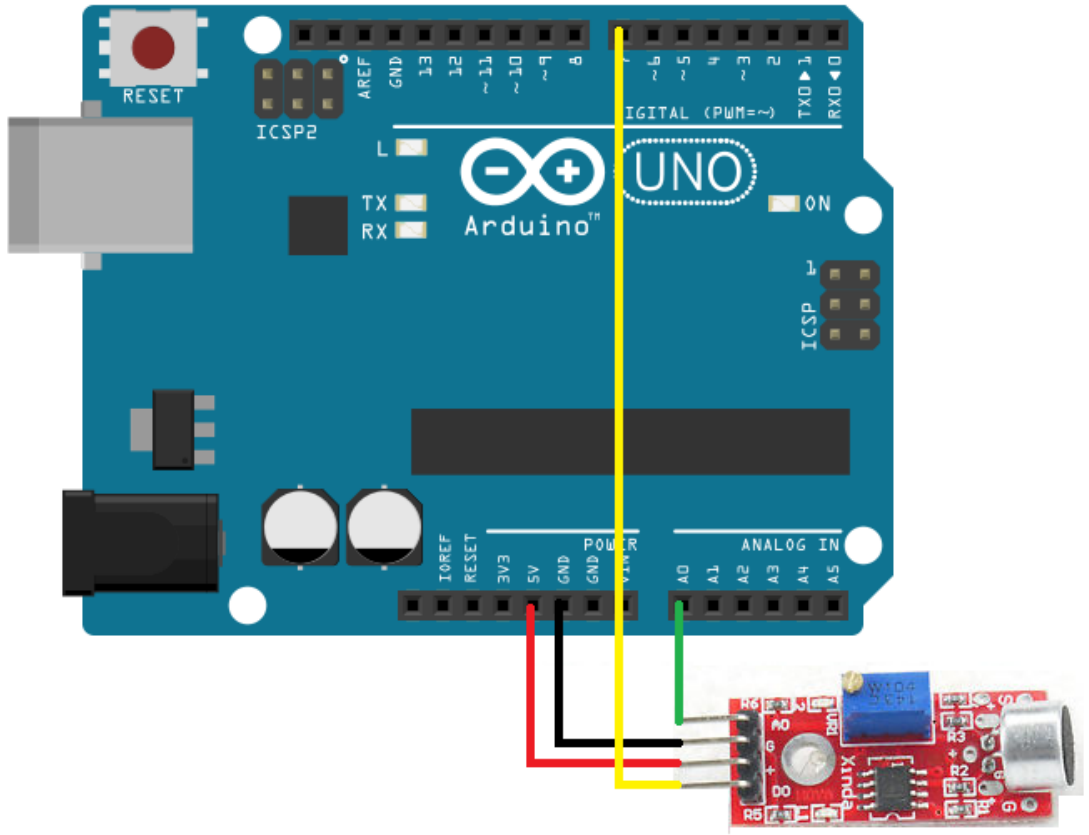


- 단자 설명

Parameter	Value
VCC	5 Vdc from your Arduino
Ground	GND from your Arduino
A0	Connect to Analog Input Pin
D0	Connect to Digital Input Pin
Power LED	Illuminates when power is applied
Sound Detection LED	Illuminates when sound is detected
Sound Set Point Adjust	CW = More Sensitive CCW = Less Sensitive

Sound sensor

- 결선



- 소리 입력 상태 출력

5-3-1.ino

```
#define SAIN  A0
#define SDIN  7

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

void loop() {
  bool sd = digitalRead(SDIN);
  int  sa = analogRead(SAIN);
  Serial.print("digital: ");
  Serial.println(sd, DEC);
  Serial.print("analog: ");
  Serial.println(sa, DEC);
  delay(300);
}
```

Sound sensor

- 박수 두 번으로 #13 LED 토글 (delay 이용 버전)

5-3-2.ino

```
#define SND A0
#define LED 13
#define THREADSHOLD 100

int snd_count = 0;

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

void loop() {
  read_sound();

  if(snd_count == 2) {
    digitalWrite(LED, !digitalRead(LED));
    snd_count=0;
  }
}
```

```
void read_sound() {
  int sound = analogRead(SND);
  if(sound > THREADSHOLD) {
    snd_count++;
    Serial.print("snd_count: ");
    Serial.println(snd_count, DEC);
    // 0.2초 안에 다시 감지되지 않도록
    delay(200);
  }
}
```

Sound sensor

- 박수 두 번으로 #13 LED 토글 (멀티태스킹 가능 버전)

```
#define SND A0
#define LED 13
#define THREADSHOLD 100

int snd_count = 0;

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

void loop() {
  read_sound();

  if(snd_count == 2) {
    digitalWrite(LED, !digitalRead(LED));
    snd_count=0;
  }
}
```

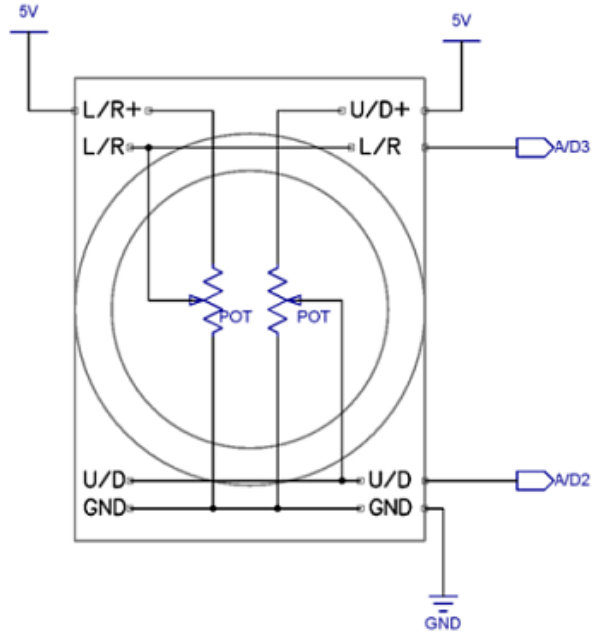
```
void read_sound() {
  static unsigned long prev_time = -200;
  if(millis() - prev_time > 500) {
    snd_count=0;
  }
  if(millis() - prev_time > 200) {
    int sound = analogRead(SND);
    //Serial.println(sound, DEC);
    if(sound > THREADSHOLD) {
      snd_count++;
      Serial.print("snd_count: ");
      Serial.println(snd_count, DEC);
      prev_time = millis();
    }
  }
}
```

조이스틱

- 외관

조이스틱 핀	역할	아두이노
VCC	전원(+)	5V
GND	전원(-)	GND
VRx	X축 입력(수평방향)	A0
VRy	Y축 입력(수직방향)	A1
SW	누름 입력	A2

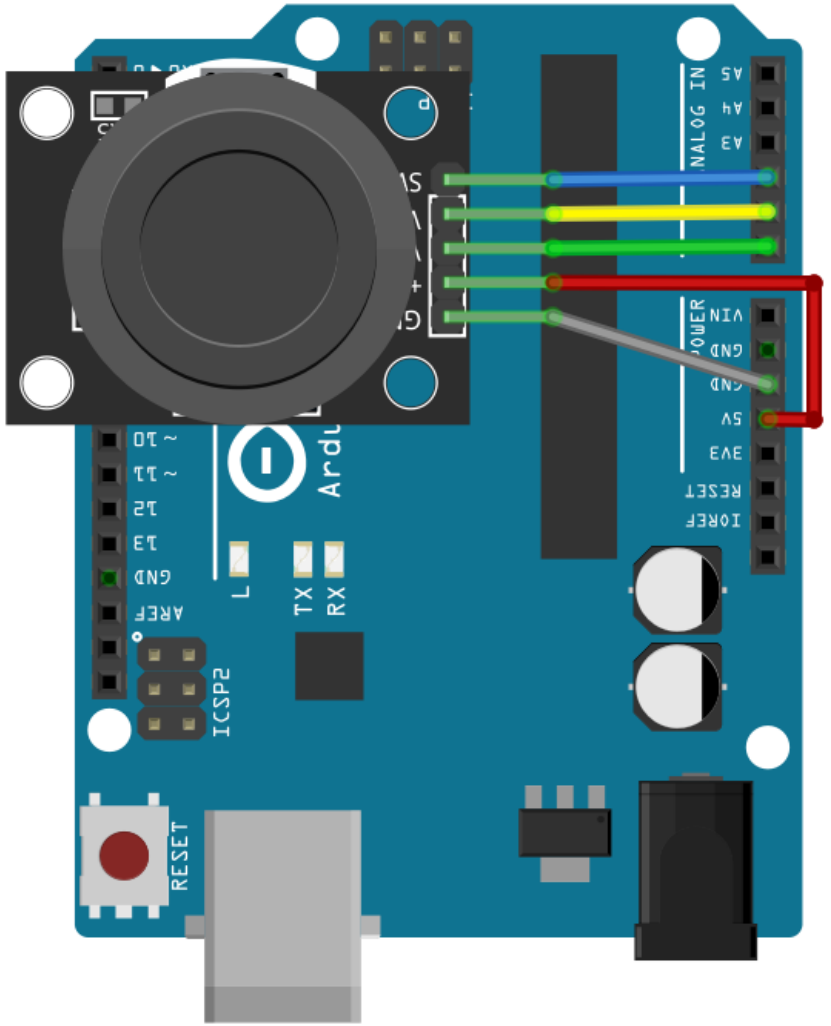
- 조이스틱 내부



- X, Y는 가변저항에 연결되어 있음
- SW는 일반 버튼과 동일하므로 풀업 또는 풀다운 저항처리 해야 함.

조이스틱

- 결선



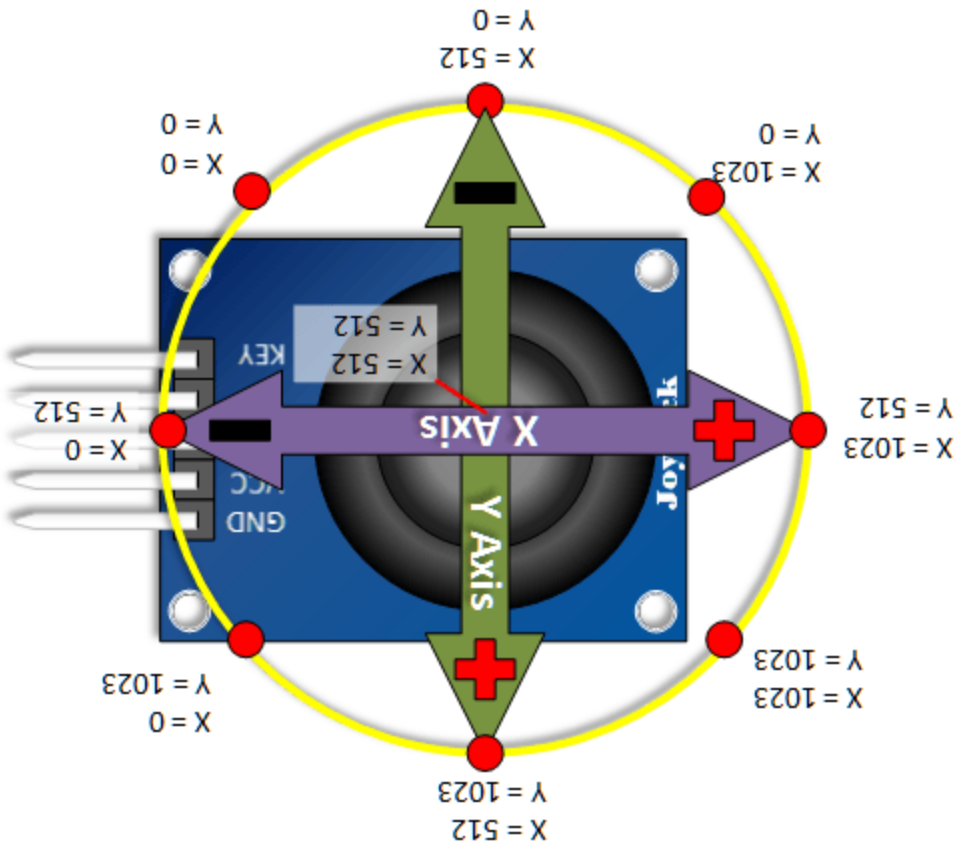
- 회로 연결

조이스틱 핀	역할	아두이노
VCC	전원(+)	5V
GND	전원(-)	GND
VRx	X축 입력(수평방향)	A0
VRy	Y축 입력(수직방향)	A1
SW	누름 입력	A2

조이스틱

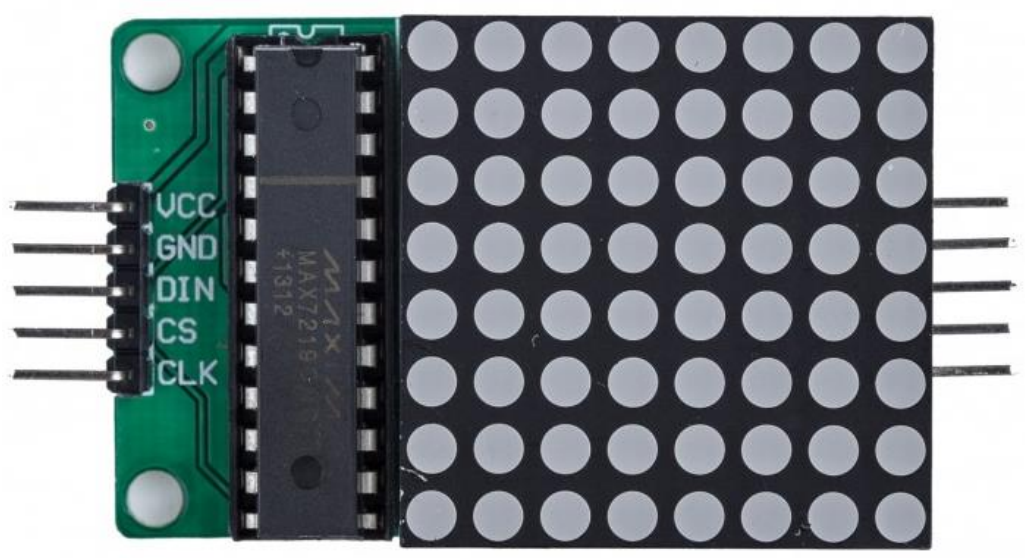
5-4-1.ino

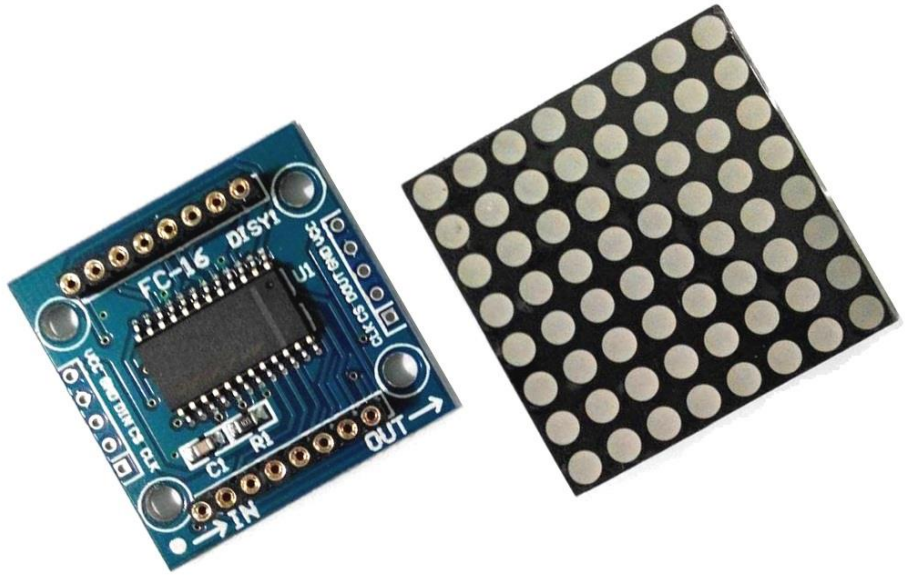
```
#define VRX A0
#define VRY A1
#define SW  A2
//SW는 디지털입력이지만 아날로그핀에 연결해도 무방

void setup() {
  pinMode(SW, INPUT_PULLUP);
  Serial.begin(9600);
}

void loop() {
  int hori = analogRead(VRX);
  int vert = analogRead(VRY);
  int btn = digitalRead(SW);

  Serial.print("hori: ");
  Serial.println(hori);
  Serial.print("vert: ");
  Serial.println(vert);
  Serial.print("btn: ");
  Serial.println(btn);
  delay(250);
}
```



8 x 8 dot matrix using

Serially Interfaced, 8-Digit LED Display Drivers (MAX7219/7221)

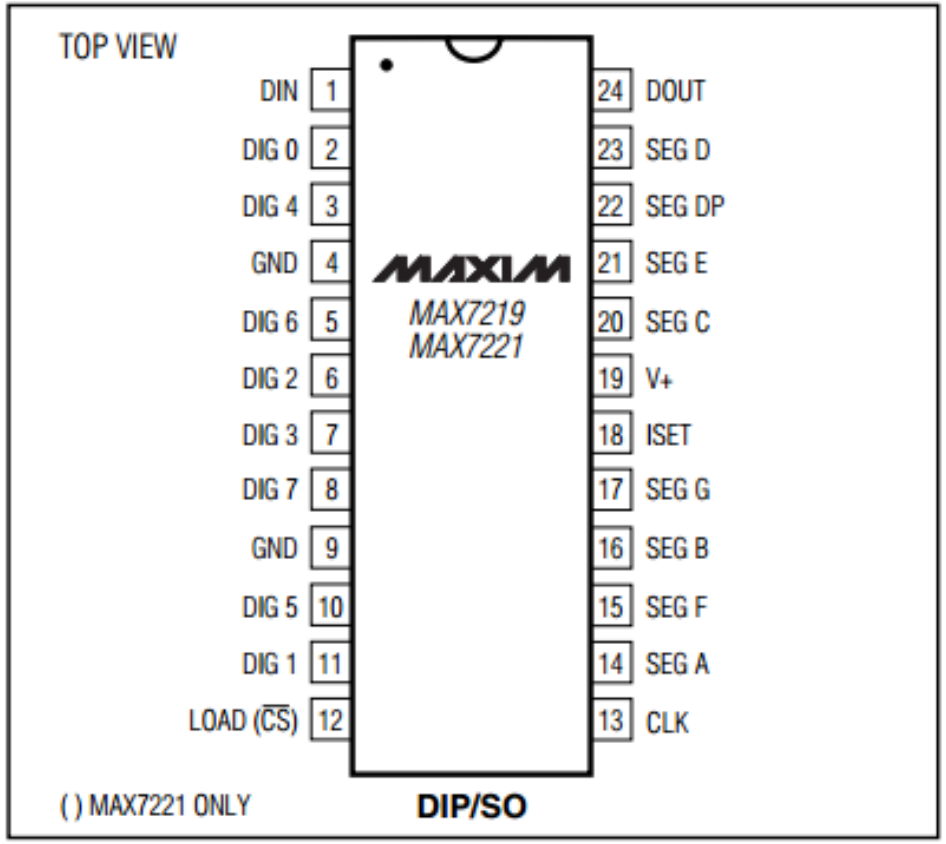




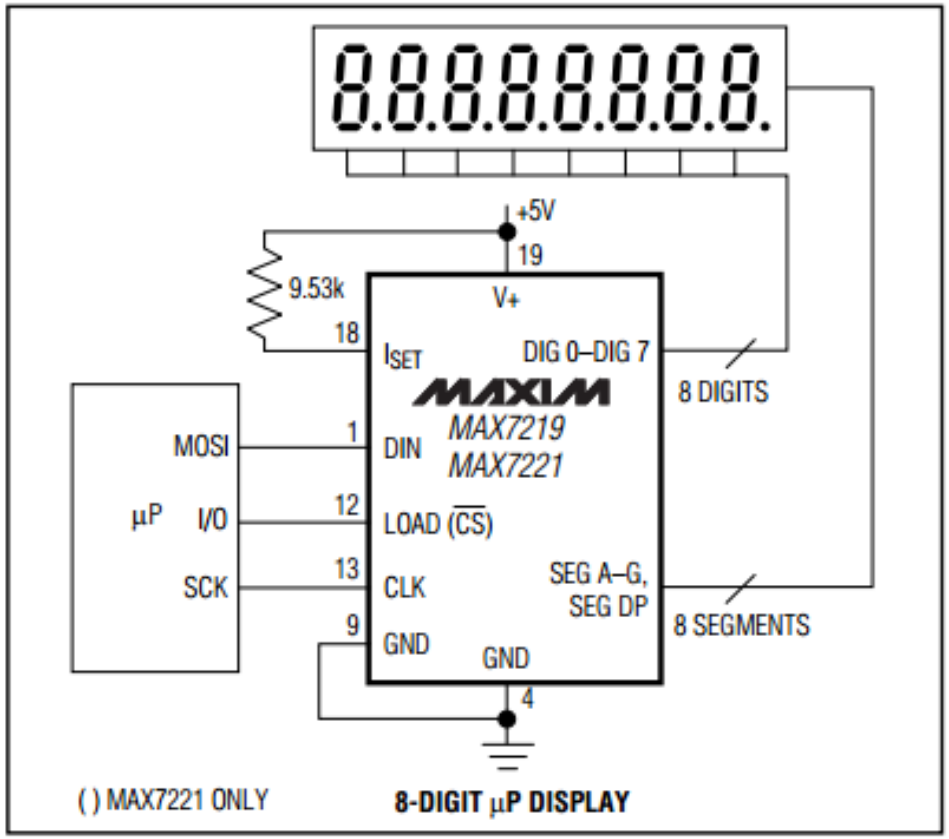
8 x 8 dot matrix using

Serially Interfaced, 8-Digit LED Display Drivers (MAX7219/7221)

Pin Configuration



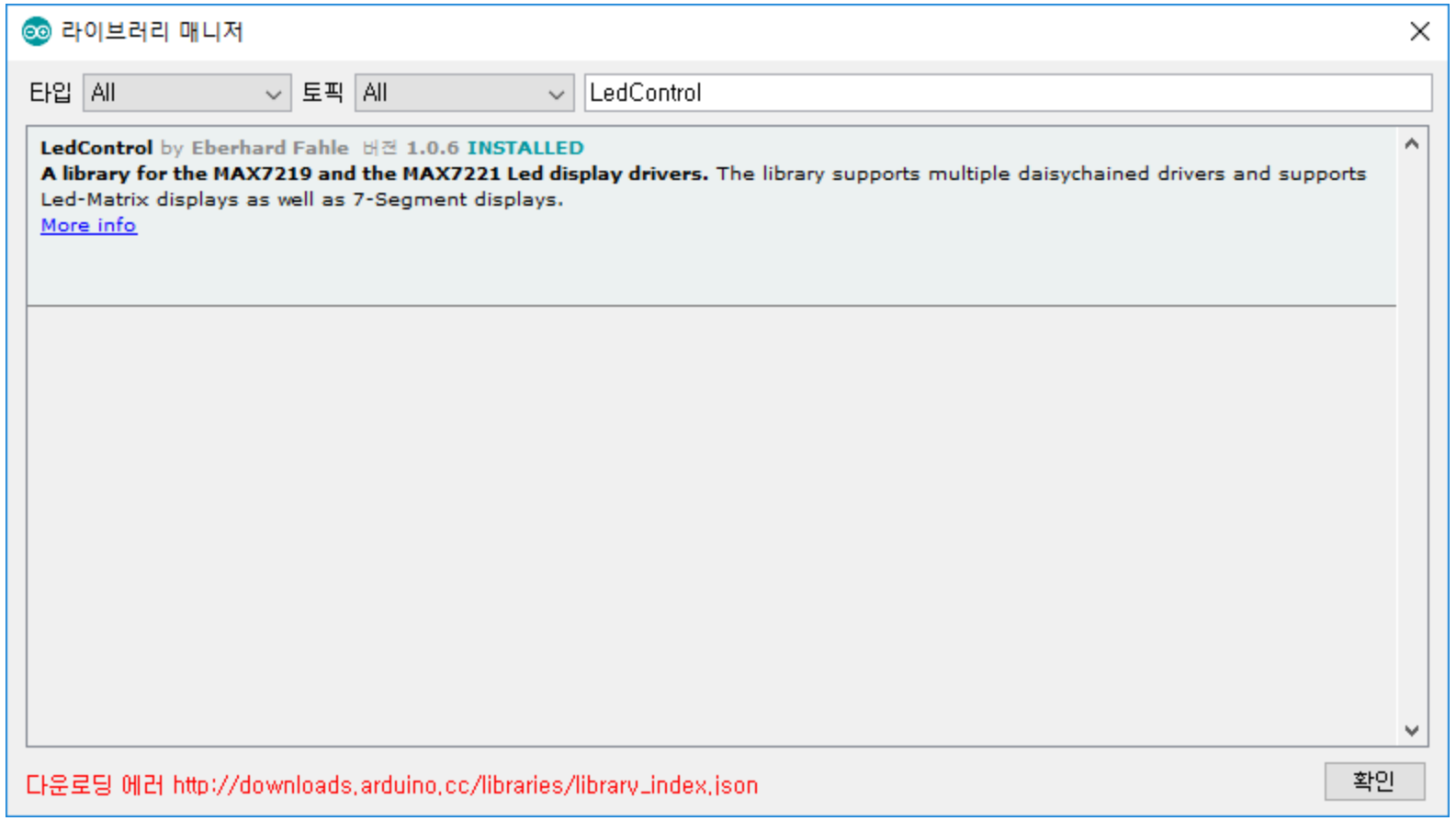
Typical Application Circuit



8 x 8 dot matrix using

Serially Interfaced, 8-Digit LED Display Drivers (MAX7219/7221)

- LedControl 라이브러리 다운로드



LedControl Library

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

- 장치 초기화
  - **LedControl(dataPin, clockPin, csPin, numDevices);**
- 디스플레이 밝기 조절
  - **setIntensity(int addr, int intensity);**
  - 큰 숫자가 디스플레이를 더 밝게 만듦. 15가 최대치
- 디스플레이 지우기
  - **clearDisplay(addr);**
- 전원 절약모드 설정
  - **void shutdown(int addr, bool b);**

Controlling a Led matrix

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

- LED 한 줄 제어(행 단위)
  - **void setRow(int addr, int row, byte value);**

- LED 한 줄 제어(열 단위)
  - **void setColumn(int addr, int col, byte value);**
  - 내부적으로 구현이 setRow를 8번 호출하는 형태이므로 느림.

- LED 픽셀 단위로 제어
  - **void setLed(int addr, int row, int col, boolean state);**

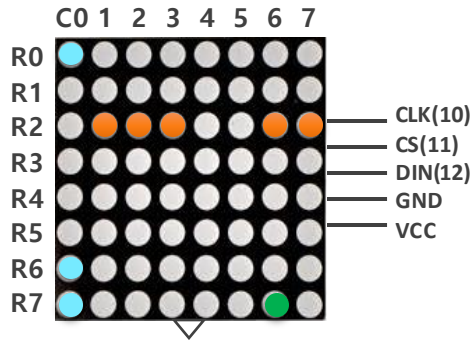
Controlling a Led matrix

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

- 샘플코드

5-5-1.ino

```
#include <LedControl.h>

#define CLK 10
#define CS  11
#define DIN 12
#define BIRGHTNESS 8

LedControl lc = LedControl(DIN, CLK, CS,1);

void setup() {
  lc.shutdown(0, false);
  lc.setIntensity(0, BIRGHTNESS);
  lc.clearDisplay(0);

  lc.setRow(0, 2, B01110011);
  lc.setColumn(0, 0, B10000011);
  lc.setLed(0, 7, 6, true);
}

void loop() {}
```

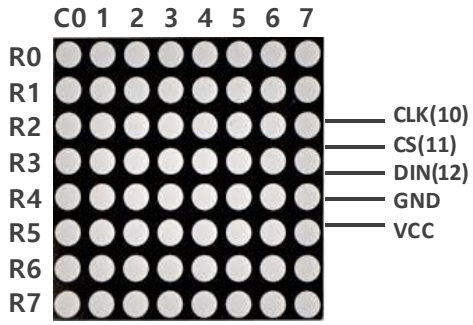


8 x 8 dot matrix using

Serially Interfaced, 8-Digit LED Display Drivers (MAX7219/7221)

5-5-2.ino

```
#include <LedControl.h>

#define DIN 12
#define CS  11
#define CLK 10

LedControl lc=LedControl(DIN, CLK, CS,1);

void setup() {
  // the zero refers to the MAX7219 number, it is zero for 1 chip
  lc.shutdown(0,false);   // turn off power saving, enables display
  lc.setIntensity(0,1);   // sets brightness (0~15 possible values)
  lc.clearDisplay(0);     // clear screen
}

void loop() {
  for (int row=0; row<8; row++) {
    for (int col=0; col<8; col++) {
      lc.setLed(0,row,col,true); // turns on LED at col, row
      delay(25);
    }
  }

  for (int row=0; row<8; row++) {
    for (int col=0; col<8; col++)  {
      lc.setLed(0,row,col,false); // turns off LED at col, row
      delay(25);
    }
  }
}
```



8 x 8 Font of ASCII

font.h

```
const byte font[128][8] = {
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0000 (nul)
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0001
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0002
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0003
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0004
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0005
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0006
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0007
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0008
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0009
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+000A
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+000B
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+000C
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+000D
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+000E
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+000F
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0010
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0011
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0012
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0013
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0014
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0015
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0016
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0017
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0018
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0019
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+001A
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+001B
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+001C
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+001D
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+001E
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+001F
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0020 (space)
  { 0x18, 0x3C, 0x3C, 0x18, 0x18, 0x00, 0x18, 0x00 },   // U+0021 (!)
  { 0x6C, 0x6C, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0022 (")
  { 0x6C, 0x6C, 0xFE, 0x6C, 0xFE, 0x6C, 0x6C, 0x00 },   // U+0023 (#)
  { 0x30, 0x7C, 0xC0, 0x78, 0x0C, 0xF8, 0x30, 0x00 },   // U+0024 ($)
  { 0x00, 0xC6, 0xCC, 0x18, 0x30, 0x66, 0xC6, 0x00 },   // U+0025 (%)
  { 0x38, 0x6C, 0x38, 0x76, 0xDC, 0xCC, 0x76, 0x00 },   // U+0026 (&)
  { 0x60, 0x60, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0027 (')
  { 0x18, 0x30, 0x60, 0x60, 0x60, 0x30, 0x18, 0x00 },   // U+0028 (()
  { 0x60, 0x30, 0x18, 0x18, 0x18, 0x30, 0x60, 0x00 },   // U+0029 ())
  { 0x00, 0x66, 0x3C, 0xFF, 0x3C, 0x66, 0x00, 0x00 },   // U+002A (*)
  { 0x00, 0x30, 0x30, 0xFC, 0x30, 0x30, 0x00, 0x00 },   // U+002B (+)
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x30, 0x30, 0x60 },   // U+002C (,)
  { 0x00, 0x00, 0x00, 0xFC, 0x00, 0x00, 0x00, 0x00 },   // U+002D (-)
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x30, 0x30, 0x00 },   // U+002E (.)
  { 0x06, 0x0C, 0x18, 0x30, 0x60, 0xC0, 0x80, 0x00 },   // U+002F (/)
  { 0x7C, 0xC6, 0xCE, 0xDE, 0xF6, 0xE6, 0x7C, 0x00 },   // U+0030 (0)
  { 0x30, 0x70, 0x30, 0x30, 0x30, 0x30, 0xFC, 0x00 },   // U+0031 (1)
  { 0x78, 0xCC, 0x0C, 0x38, 0x60, 0xCC, 0xFC, 0x00 },   // U+0032 (2)
  { 0x78, 0xCC, 0x0C, 0x38, 0x0C, 0xCC, 0x78, 0x00 },   // U+0033 (3)
  { 0x1C, 0x3C, 0x6C, 0xCC, 0xFE, 0x0C, 0x1E, 0x00 },   // U+0034 (4)
  { 0xFC, 0xC0, 0xF8, 0x0C, 0x0C, 0xCC, 0x78, 0x00 },   // U+0035 (5)
  { 0x38, 0x60, 0xC0, 0xF8, 0xCC, 0xCC, 0x78, 0x00 },   // U+0036 (6)
  { 0xFC, 0xCC, 0x0C, 0x18, 0x30, 0x30, 0x30, 0x00 },   // U+0037 (7)
  { 0x78, 0xCC, 0xCC, 0x78, 0xCC, 0xCC, 0x78, 0x00 },   // U+0038 (8)
  { 0x78, 0xCC, 0xCC, 0x7C, 0x0C, 0x18, 0x70, 0x00 },   // U+0039 (9)
  { 0x00, 0x30, 0x30, 0x00, 0x00, 0x30, 0x30, 0x00 },   // U+003A (:)
  { 0x00, 0x30, 0x30, 0x00, 0x00, 0x30, 0x30, 0x60 },   // U+003B (//)
  { 0x18, 0x30, 0x60, 0xC0, 0x60, 0x30, 0x18, 0x00 },   // U+003C (<)
  { 0x00, 0x00, 0xFC, 0x00, 0x00, 0xFC, 0x00, 0x00 },   // U+003D (=)
  { 0x60, 0x30, 0x18, 0x0C, 0x18, 0x30, 0x60, 0x00 },   // U+003E (>)
  { 0x78, 0xCC, 0x0C, 0x18, 0x30, 0x00, 0x30, 0x00 },   // U+003F (?)
  { 0x7C, 0xC6, 0xDE, 0xDE, 0xDE, 0xC0, 0x78, 0x00 },   // U+0040 (@)
  { 0x30, 0x78, 0xCC, 0xCC, 0xFC, 0xCC, 0xCC, 0x00 },   // U+0041 (A)
  { 0xFC, 0x66, 0x66, 0x7C, 0x66, 0x66, 0xFC, 0x00 },   // U+0042 (B)
  { 0x3C, 0x66, 0xC0, 0xC0, 0xC0, 0x66, 0x3C, 0x00 },   // U+0043 (C)
  { 0xF8, 0x6C, 0x66, 0x66, 0x66, 0x6C, 0xF8, 0x00 },   // U+0044 (D)
  { 0xFE, 0x62, 0x68, 0x78, 0x68, 0x62, 0xFE, 0x00 },   // U+0045 (E)
  { 0xFE, 0x62, 0x68, 0x78, 0x68, 0x60, 0xF0, 0x00 },   // U+0046 (F)
  { 0x3C, 0x66, 0xC0, 0xC0, 0xCE, 0x66, 0x3E, 0x00 },   // U+0047 (G)
  { 0xCC, 0xCC, 0xCC, 0xFC, 0xCC, 0xCC, 0xCC, 0x00 },   // U+0048 (H)
  { 0x78, 0x30, 0x30, 0x30, 0x30, 0x30, 0x78, 0x00 },   // U+0049 (I)
  { 0x1E, 0x0C, 0x0C, 0x0C, 0xCC, 0xCC, 0x78, 0x00 },   // U+004A (J)
  { 0xE6, 0x66, 0x6C, 0x78, 0x6C, 0x66, 0xE6, 0x00 },   // U+004B (K)
  { 0xF0, 0x60, 0x60, 0x60, 0x62, 0x66, 0xFE, 0x00 },   // U+004C (L)
  { 0xC6, 0xEE, 0xFE, 0xFE, 0xD6, 0xC6, 0xC6, 0x00 },   // U+004D (M)
  { 0xC6, 0xE6, 0xF6, 0xDE, 0xCE, 0xC6, 0xC6, 0x00 },   // U+004E (N)
  { 0x38, 0x6C, 0xC6, 0xC6, 0xC6, 0x6C, 0x38, 0x00 },   // U+004F (O)
  { 0xFC, 0x66, 0x66, 0x7C, 0x60, 0x60, 0xF0, 0x00 },   // U+0050 (P)
  { 0x78, 0xCC, 0xCC, 0xCC, 0xDC, 0x78, 0x1C, 0x00 },   // U+0051 (Q)
  { 0xFC, 0x66, 0x66, 0x7C, 0x6C, 0x66, 0xE6, 0x00 },   // U+0052 (R)
  { 0x78, 0xCC, 0xE0, 0x70, 0x1C, 0xCC, 0x78, 0x00 },   // U+0053 (S)
  { 0xFC, 0xB4, 0x30, 0x30, 0x30, 0x30, 0x78, 0x00 },   // U+0054 (T)
  { 0xCC, 0xCC, 0xCC, 0xCC, 0xCC, 0xCC, 0xFC, 0x00 },   // U+0055 (U)
  { 0xCC, 0xCC, 0xCC, 0xCC, 0xCC, 0x78, 0x30, 0x00 },   // U+0056 (V)
  { 0xC6, 0xC6, 0xC6, 0xD6, 0xFE, 0xEE, 0xC6, 0x00 },   // U+0057 (W)
  { 0xC6, 0xC6, 0x6C, 0x38, 0x38, 0x6C, 0xC6, 0x00 },   // U+0058 (X)
  { 0xCC, 0xCC, 0xCC, 0x78, 0x30, 0x30, 0x78, 0x00 },   // U+0059 (Y)
  { 0xFE, 0xC6, 0x8C, 0x18, 0x32, 0x66, 0xFE, 0x00 },   // U+005A (Z)
  { 0x78, 0x60, 0x60, 0x60, 0x60, 0x60, 0x78, 0x00 },   // U+005B ([)
  { 0xC0, 0x60, 0x30, 0x18, 0x0C, 0x06, 0x02, 0x00 },   // U+005C (\)
  { 0x78, 0x18, 0x18, 0x18, 0x18, 0x18, 0x78, 0x00 },   // U+005D (])
  { 0x10, 0x38, 0x6C, 0xC6, 0x00, 0x00, 0x00, 0x00 },   // U+005E (^)
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF },   // U+005F (_)
  { 0x30, 0x30, 0x18, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+0060 (`)
  { 0x00, 0x00, 0x78, 0x0C, 0x7C, 0xCC, 0x76, 0x00 },   // U+0061 (a)
  { 0xE0, 0x60, 0x60, 0x7C, 0x66, 0x66, 0xDC, 0x00 },   // U+0062 (b)
  { 0x00, 0x00, 0x78, 0xCC, 0xC0, 0xCC, 0x78, 0x00 },   // U+0063 (c)
  { 0x1C, 0x0C, 0x0C, 0x7C, 0xCC, 0xCC, 0x76, 0x00 },   // U+0064 (d)
  { 0x00, 0x00, 0x78, 0xCC, 0xFC, 0xC0, 0x78, 0x00 },   // U+0065 (e)
  { 0x38, 0x6C, 0x60, 0xF0, 0x60, 0x60, 0xF0, 0x00 },   // U+0066 (f)
  { 0x00, 0x00, 0x76, 0xCC, 0xCC, 0x7C, 0x0C, 0xF8 },   // U+0067 (g)
  { 0xE0, 0x60, 0x6C, 0x76, 0x66, 0x66, 0xE6, 0x00 },   // U+0068 (h)
  { 0x30, 0x00, 0x70, 0x30, 0x30, 0x30, 0x78, 0x00 },   // U+0069 (i)
  { 0x0C, 0x00, 0x0C, 0x0C, 0x0C, 0xCC, 0xCC, 0x78 },   // U+006A (j)
  { 0xE0, 0x60, 0x66, 0x6C, 0x78, 0x6C, 0xE6, 0x00 },   // U+006B (k)
  { 0x70, 0x30, 0x30, 0x30, 0x30, 0x30, 0x78, 0x00 },   // U+006C (l)
  { 0x00, 0x00, 0xCC, 0xFE, 0xFE, 0xD6, 0xC6, 0x00 },   // U+006D (m)
  { 0x00, 0x00, 0xF8, 0xCC, 0xCC, 0xCC, 0xCC, 0x00 },   // U+006E (n)
  { 0x00, 0x00, 0x78, 0xCC, 0xCC, 0xCC, 0x78, 0x00 },   // U+006F (o)
  { 0x00, 0x00, 0xDC, 0x66, 0x66, 0x7C, 0x60, 0xF0 },   // U+0070 (p)
  { 0x00, 0x00, 0x76, 0xCC, 0xCC, 0x7C, 0x0C, 0x1E },   // U+0071 (q)
  { 0x00, 0x00, 0xDC, 0x76, 0x66, 0x60, 0xF0, 0x00 },   // U+0072 (r)
  { 0x00, 0x00, 0x7C, 0xC0, 0x78, 0x0C, 0xF8, 0x00 },   // U+0073 (s)
  { 0x10, 0x30, 0x7C, 0x30, 0x30, 0x34, 0x18, 0x00 },   // U+0074 (t)
  { 0x00, 0x00, 0xCC, 0xCC, 0xCC, 0xCC, 0x76, 0x00 },   // U+0075 (u)
  { 0x00, 0x00, 0xCC, 0xCC, 0xCC, 0x78, 0x30, 0x00 },   // U+0076 (v)
  { 0x00, 0x00, 0xC6, 0xD6, 0xFE, 0xFE, 0x6C, 0x00 },   // U+0077 (w)
  { 0x00, 0x00, 0xC6, 0x6C, 0x38, 0x6C, 0xC6, 0x00 },   // U+0078 (x)
  { 0x00, 0x00, 0xCC, 0xCC, 0xCC, 0x7C, 0x0C, 0xF8 },   // U+0079 (y)
  { 0x00, 0x00, 0xFC, 0x98, 0x30, 0x64, 0xFC, 0x00 },   // U+007A (z)
  { 0x1C, 0x30, 0x30, 0xE0, 0x30, 0x30, 0x1C, 0x00 },   // U+007B ({)
  { 0x18, 0x18, 0x18, 0x00, 0x18, 0x18, 0x18, 0x00 },   // U+007C (|)
  { 0xE0, 0x30, 0x30, 0x1C, 0x30, 0x30, 0xE0, 0x00 },   // U+007D (})
  { 0x76, 0xDC, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+007E (~)
  { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 },   // U+007F
};
```

8 x 8 Font of ASCII

- 폰트 자료구조

```
font[ 72][8] = { 0xCC, 0xCC, 0xCC, 0xFC, 0xCC, 0xCC, 0xCC, 0x00 };
font[101][8] = { 0x00, 0x00, 0x78, 0xCC, 0xFC, 0xC0, 0x78, 0x00 };
font[108][8] = { 0x70, 0x30, 0x30, 0x30, 0x30, 0x30, 0x78, 0x00 };
```

```
1100 1100    0000 0000    0111 0000
1100 1100    0000 0000    0011 0000      . . .
1100 1100    0111 1000    0011 0000      . . .
1111 1100    1100 1100    0011 0000      . . .
1100 1100    1111 1100    0011 0000      . . .
1100 1100    1100 0000    0011 0000      . . .
1100 1100    0111 1000    0111 1000      . . .
0000 0000    0000 0000    0000 0000      . . .
```

Ctrl	Dec	Hex	Char	Code
^@	0	00		NUL
^A	1	01		SOH
^B	2	02		STX
^C	3	03		ETX
^D	4	04		EOT
^E	5	05		ENQ
^F	6	06		ACK
^G	7	07		BEL
^H	8	08		BS
^I	9	09		HT
^J	10	0A		LF
^K	11	0B		VT
^L	12	0C		FF
^M	13	0D		CR
^N	14	0E		SO
^O	15	0F		SI
^P	16	10		DLE
^Q	17	11		DC1
^R	18	12		DC2
^S	19	13		DC3
^T	20	14		DC4
^U	21	15		NAK
^V	22	16		SYN
^W	23	17		ETB
^X	24	18		CAN
^Y	25	19		EM
^Z	26	1A		SUB
^[	27	1B		ESC
^\	28	1C		FS
^]	29	1D		GS
^^	30	1E	▲	RS
^-	31	1F	▼	US

Dec	Hex	Char
32	20	
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?

Dec	Hex	Char
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_

Dec	Hex	Char
96	60	`
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	⌂

* ASCII code 127 has the code DEL. Under MS-DOS, this code has the same effect as ASCII 8 (BS). The DEL code can be generated by the CTRL + BKSP key.

8 x 8 dot matrix 글자 출력

- 실습과제
  - 다음 페이지의 font 정보 활용하여 0~9, A~Z, a~z 까지 문자를 순차적으로 출력하는 프로그램을 작성하시오.

- 소스코드

```
#include "font.h"
#include <LedControl.h>

#define CLK 10
#define CS  11
#define DIN 12
#define BIRGHTNESS 8

LedControl lc = LedControl(DIN, CLK, CS,1);

void setup() {
  lc.shutdown(0, false);
  lc.setIntensity(0, BIRGHTNESS);
  lc.clearDisplay(0);
}

void loop() {
  //
}
```

8 x 8 dot matrix 글자 출력

- 퀴즈 정답 소스코드

5-5-3.ino

```
#include "font.h"
#include <LedControl.h>

#define CLK 10
#define CS  11
#define DIN 12
#define BIRGHTNESS 8

LedControl lc = LedControl(DIN, CLK, CS,1);

void setup() {
  lc.shutdown(0, false);
  lc.setIntensity(0, BIRGHTNESS);
  lc.clearDisplay(0);
}
```

```
void loop() {
  for(int c='0'; c<='z'; c++) {
    for(int l=0; l<8; l++) {
      lc.setRow(0, l, font[c][l]);
    }
    delay(500);
  }
}
```

ASCII Table

Ctrl	Dec	Hex	Char	Code
^@	0	00		NUL
^A	1	01		SOH
^B	2	02		STX
^C	3	03		ETX
^D	4	04		EOT
^E	5	05		ENQ
^F	6	06		ACK
^G	7	07		BEL
^H	8	08		BS
^I	9	09		HT
^J	10	0A		LF
^K	11	0B		VT
^L	12	0C		FF
^M	13	0D		CR
^N	14	0E		SO
^O	15	0F		SI
^P	16	10		DLE
^Q	17	11		DC1
^R	18	12		DC2
^S	19	13		DC3
^T	20	14		DC4
^U	21	15		NAK
^V	22	16		SYN
^W	23	17		ETB
^X	24	18		CAN
^Y	25	19		EM
^Z	26	1A		SUB
^[	27	1B		ESC
^\	28	1C		FS
^]	29	1D		GS
^^	30	1E	▲	RS
^-	31	1F	▼	US

Dec	Hex	Char
32	20	
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?

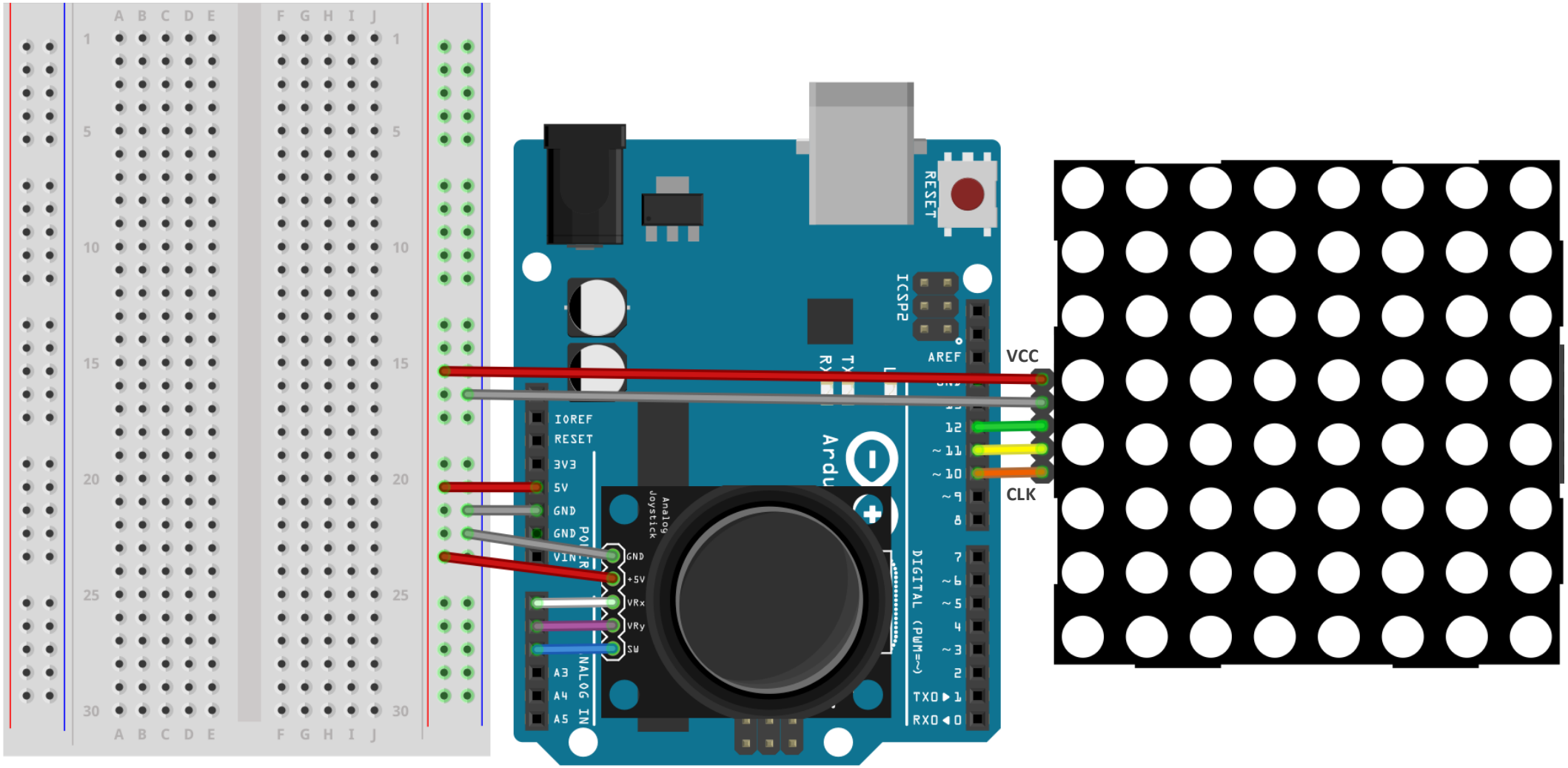
Dec	Hex	Char
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_

Dec	Hex	Char
96	60	`
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	⌂

* ASCII code 127 has the code DEL. Under MS-DOS, this code has the same effect as ASCII 8 (BS). The DEL code can be generated by the CTRL + BKSP key.

그림판

(8x8 도트 매트릭스 x 조이스틱)



그림판

(8x8 도트 매트릭스 x 조이스틱)

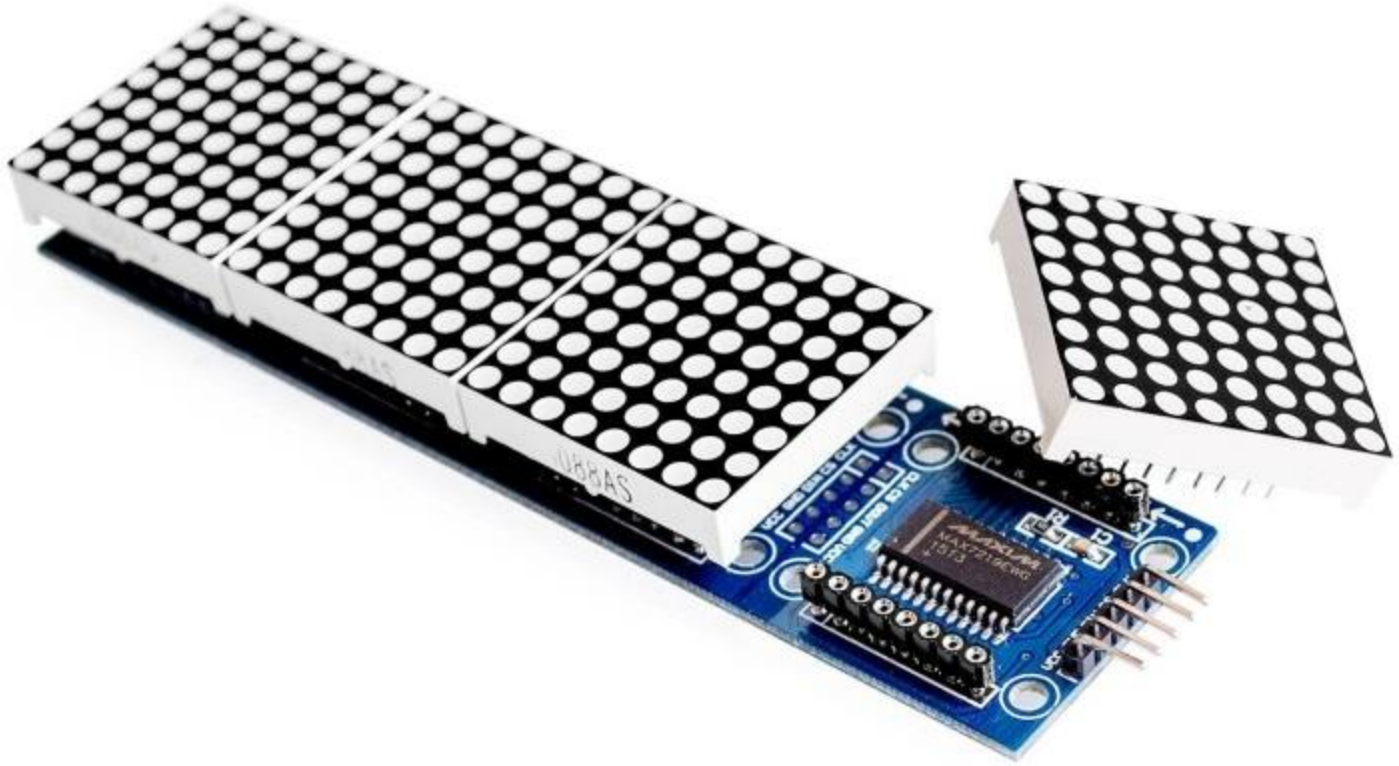
5-5-4.ino

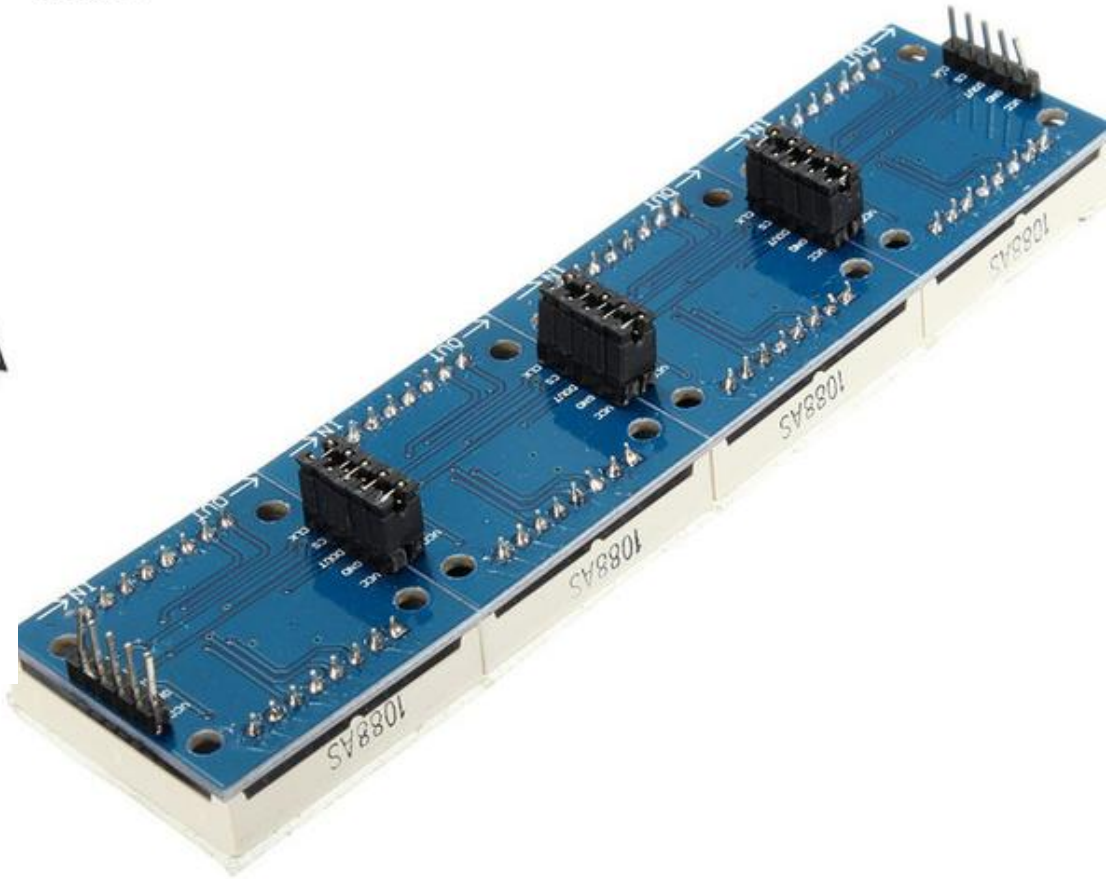
```
#include <LedControl.h>

#define CLK 10
#define CS  11
#define DIN 12
#define BIRGHTNESS 8

#define VRX A0
#define VRY A1
#define VSW A2
#define STEP 64

LedControl lc = LedControl(DIN, CLK, CS,1);
byte lc_buffer[8][8] = {0, };
int  cx = 3, cy = 3;

void setup() {
  pinMode(VSW, INPUT_PULLUP);
  lc.shutdown(0, false);
  lc.setIntensity(0, BIRGHTNESS);
  lc.clearDisplay(0);
  Serial.begin(9600);
}
```

```
void refresh() {
  for(int y=0; y<8; y++)
    for(int x=0; x<8; x++)
      lc.setLed(0, y, x, lc_buffer[y][x]);

  lc.setLed(0, cy, cx, true);  //커서 그리기
  delay(100);
}

void loop() {
  int vx = analogRead(VRX)/STEP;
  int vy = analogRead(VRY)/STEP;

  if(vx<=3 && cx>0)        cx--;
  else if(vx>=12 && cx<7)  cx++;

  if(vy<=3 && cy>0)        cy--;
  else if(vy>=12 && cy<7)  cy++;

  if(digitalRead(VSW) == LOW) {
    lc_buffer[cy][cx] = !lc_buffer[cy][cx];
    delay(50);
  }
  refresh();
}
```

8 x 8 dot matrix using

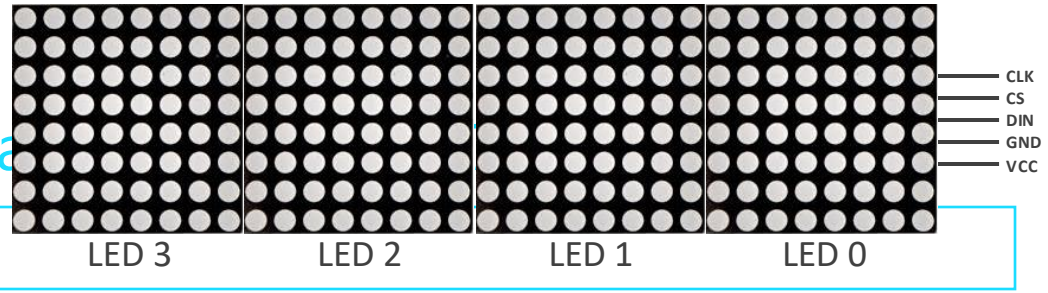
Serially Interfaced, 8-Digit LED Display Drivers (MAX7219/7221)





8 x 8 dot matrix using

Serially Interfaced, 8-Digit LED Displa

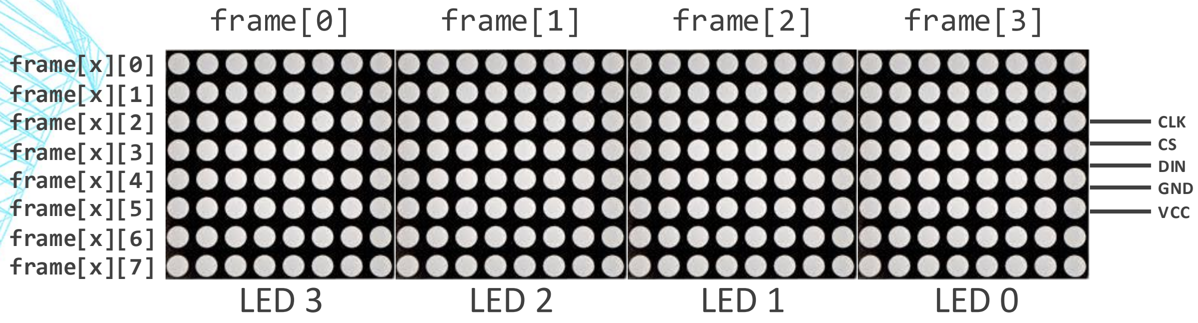


8 x 8 matrix 4개를 연결

```
#include "font.h"
#include "LedControl.h"

#define CLK 10
#define CS  11
#define DIN 12

#define BIRGHTNESS 8
#define COUNT 4

LedControl lc = LedControl(DIN, CLK, CS, COUNT);

void setup()
{
  for(int c=0; c<COUNT; c++)
  {
    lc.shutdown(c, false);
    lc.setIntensity(c, BIRGHTNESS);
    lc.setIntensity(c, 2);
    lc.clearDisplay(c);
  }
}
```

```
// 오른쪽 모듈부터 출력
void loop() {
  for(int c='0'; c<='z'; c++) {   // 글자
    for(int n=0; n<COUNT; n++) {  // LED
      for(int l=0; l<8; l++) {    // line
        lc.setRow(n, l, font[c+n][l]);
      }
    }
    delay(500);
  }
}
// 왼쪽 모듈부터 출력
void loop() {
  for(int c='0'; c<='z'; c++) {
    for(int n=COUNT-1; n>=0; n--) {
      for(int l=0; l<8; l++) {
        lc.setRow(n, l, font[c+(COUNT-1-n)][l]);
      }
    }
    delay(500);
  }
}
```

8 x 8 dot matrix 로 전광판 구현



COUNT = 4

frame[0]	frame[1]	frame[2]	frame[3]	frame[4]	frame[5]	frame[6]	
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···

WIDTH = MAX + COUNT

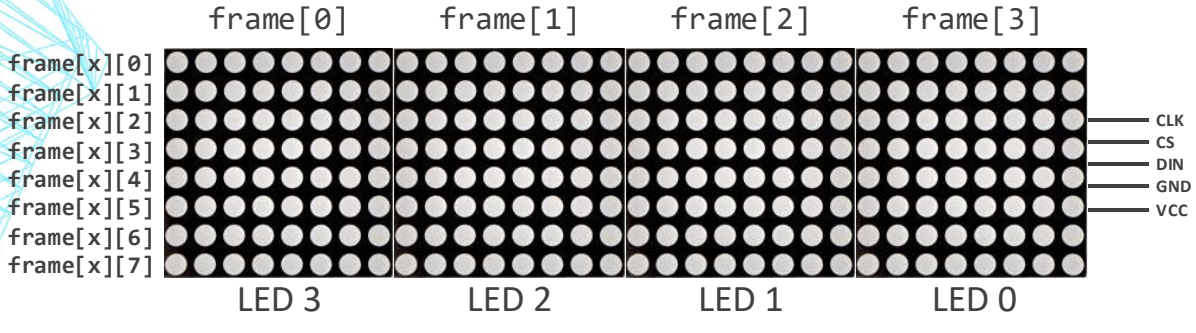
```
#include "font.h"
#include <LedControl.h>

#define CLK 10
#define CS  11
#define DIN 12

#define BIRGHTNESS 8
#define COUNT 4    // number of 8x8 matrix
#define MAX 15     // maximum string length
#define WIDTH MAX

byte frame[WIDTH][8];

LedControl lc = LedControl(DIN, CLK, CS, COUNT);
```

```
// 임시로 삼각형 모양을 그림
void makeFrame() {
  int x, y;
  for(x=0; x<WIDTH; x++) {
    byte t = 0xFF << x;
    // x=0, 1111 1111
    // x=1, 1111 1110

    for(y=0; y<8; y++) {
      frame[x][y] = t << y;
    }
  }
}

void setup() {
  for(int c=0; c<COUNT; c++) {
    lc.shutdown(c, false);
    lc.setIntensity(c, BIRGHTNESS);
    lc.clearDisplay(c);
  }
  makeFrame();
}
```

OOOOOOOO	OOOOOOO	OOOOOO	OOOOO
OOOOOOO	OOOOOO	OOOOO	OOOO
OOOOOO	OOOOO	OOOO	OOO
OOOOO	OOOO	OOO	OO
OOOO	OOO	OO	O
OOO	OO	O	
OO	O		
O			

```
void draw() {
  // LED3 🡐 frame[0], LED2 🡐 frame[1]
  // LED1 🡐 frame[2], LED0 🡐 frame[3]
  // 위와 같이 그려지도록 코딩하세요.
  for(…) {
    for(…) {
      lc.setRow(n, l, frame[][l]);
    }
  }
}

void loop() {
  draw();
  delay(80);
}
```

8 x 8 dot matrix 로 전광판 구현



frame[0]	frame[1]	frame[2]	frame[3]	frame[4]	frame[5]	frame[6]	
1100 1100	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1100 1100	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1100 1100	0111 1000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1111 1100	1100 1100	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1100 1100	1111 1100	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1100 1100	1100 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1100 1100	0111 1000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···

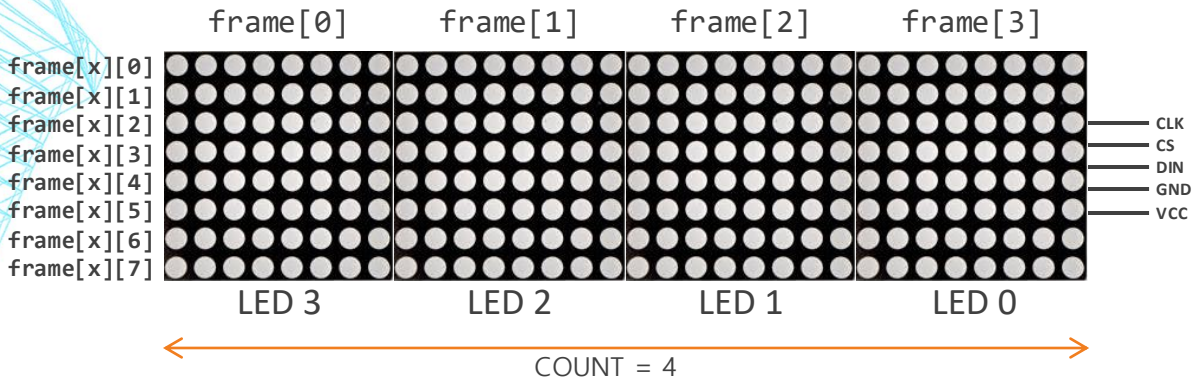
WIDTH = MAX + COUNT

```
void makeFrame(char* str) {
  int x, y;
  int len = strlen(str);

  // 파라미터로 넘어온 str문자열에서
  // 각 문자에 대한 폰트 정보를 얻어와
  // frame 버퍼에 채우는 코드를 구현하세요.
  for(x=0; x<WIDTH; x++) {
    for(y=0; y<8; y++) {
      //frame[x][y] = font[?][?];
    }
  }
}

void setup() {
  for(int c=0; c<COUNT; c++) {
    lc.shutdown(c, false);
    lc.setIntensity(c, BIRGHTNESS);
    lc.clearDisplay(c);
  }
  makeFrame("Hello~ MATRIX");
}
```

```
{ 0xCC, 0xCC, 0xCC, 0xFC, 0xCC, 0xCC, 0xCC, 0x00 },   // U+0048 (H)
{ 0x00, 0x00, 0x78, 0xCC, 0xFC, 0xC0, 0x78, 0x00 },   // U+0065 (e)
{ 0x70, 0x30, 0x30, 0x30, 0x30, 0x30, 0x78, 0x00 },   // U+006C (l)
```

```
void makeFrame(char* str) {
  int x, y;
  int len = strlen(str);

  for(x=0; x<WIDTH; x++) {
    for(y=0; y<8; y++) {
      if(x < len)
        frame[x][y] = font[str[x]][y];
      else
        frame[x][y] = 0x0;
    }
  }
}
```

8 x 8 dot matrix 로 전광판 구현



frame[0]	frame[1]	frame[2]	frame[3]	frame[4]	frame[5]	frame[6]	
1100 1100	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1100 1100	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1100 1100	0111 1000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1111 1100	1100 1100	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1100 1100	1111 1100	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1100 1100	1100 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1100 1100	0111 1000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···

frame[0]	frame[1]	frame[2]	frame[3]	frame[4]	frame[5]	frame[6]	
1001 1000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1001 1000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1001 1000	1111 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1111 1001	1001 1000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1001 1001	1111 1000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1001 1001	1000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
1001 1000	1111 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	···

```
void slideFrame() {
   // 화면이 왼쪽 방향으로 슬라이드 되도록
   // 직접 구현해 보세요
}


void loop() {
  draw();
  delay(80);
  slideFrame();
}
```

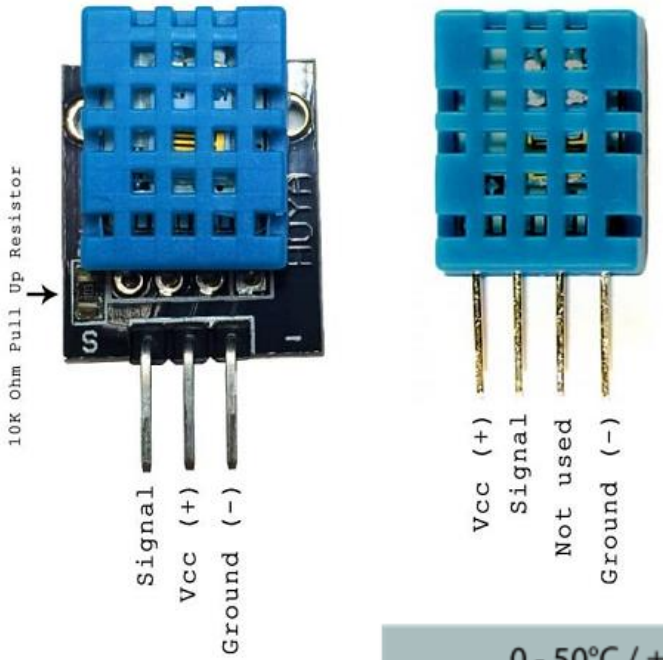
```
// 8x8 LED Matrix Drive
// by akapo@naver.com
#include <avr/pgmspace.h>
#include "font.h"
#include "LedControl.h"

#define CLK 10
#define CS  11
#define DIN 12

#define BIRGHTNESS 8
#define COUNT 4
#define MAX 40   // maximum string length
#define WIDTH (MAX+COUNT)

byte frame[WIDTH][8];

LedControl lc = LedControl(DIN, CLK, CS, COUNT);

void setup() {
  for(int c=0; c<COUNT; c++) {
    lc.shutdown(c, false);
    lc.setIntensity(c, BIRGHTNESS);
    lc.clearDisplay(c);
  }
}
```

```
void makeFrame(char* str) {
  int x, y;
  int len = strlen(str);

  for(x=0; x<WIDTH; x++) {
    for(y=0; y<8; y++) {
      if(x < COUNT)
        frame[x][y] = 0x0;
      else if(x < len+COUNT)
        frame[x][y] = font[str[x-COUNT]][y];
      else
        frame[x][y] = 0x0;
    }
  }
}

void slideFrame() {
  int x, y;
  for(y=0; y<8; y++) {
    for(x=0; x<WIDTH; x++) {
      frame[x][y] = frame[x][y]<<1;

      if(x<WIDTH-1)
        frame[x][y] |= (frame[x+1][y] & 0x80) >> 7;
    }
  }
}
```

```
char msg[4][MAX] = {
    "Hello~ MATRIX",
    "I'm neo.",
    "Keanu Reeves",
    "Wachowski Bros."
  };

void draw() {
  for(int n=COUNT-1; n>=0; n--) {
    for(int l=0; l<8; l++) {
      lc.setRow(n, l, frame[COUNT-1-n][l]);
    }
  }
}

void loop() {
  for(int m=0; m<4; m++) {
    makeFrame(msg[m]);

    int len = strlen(msg[m]);
    // COUNT+len+2 = COUNT+문자열길이+여백2글자
    for(int s=0; s<(COUNT+len+2)*8; s++) {
      draw();
      delay(40);
      slideFrame();
    }
  }
}
```

DHT11 (Humidity & Temperature Sensor)

- DHT11



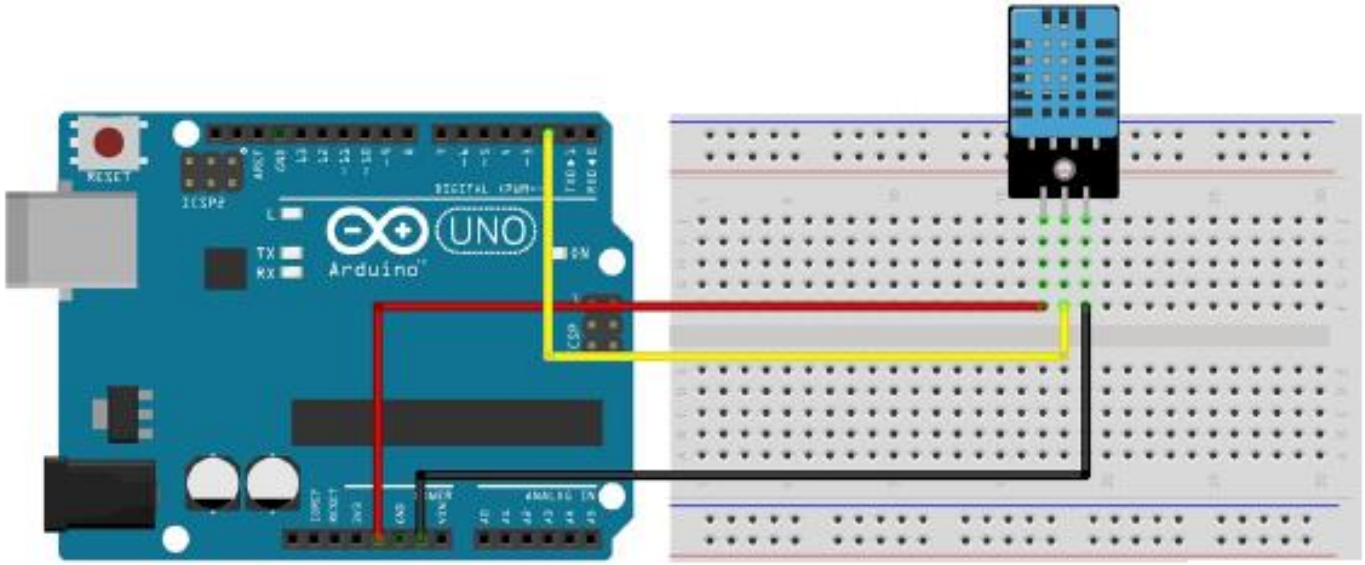
- DHT22

0 - 50°C / ± 2°C	*Temperature Range*	-40 - 125 °C / ± 0.5 °C
20 - 80% / ± 5%	*Humidity Range*	0 - 100 % / ± 2-5%
1Hz (one reading every second)	*Sampling Rate*	0.5 Hz (one reading every two seconds)
15.5mm x 12mm x 5.5mm	*Body Size*	15.1mm x 25mm x 7.7mm
3 - 5V	*Operating Voltage*	3 - 5V
2.5mA	*Max Current During Measuring*	2.5mA

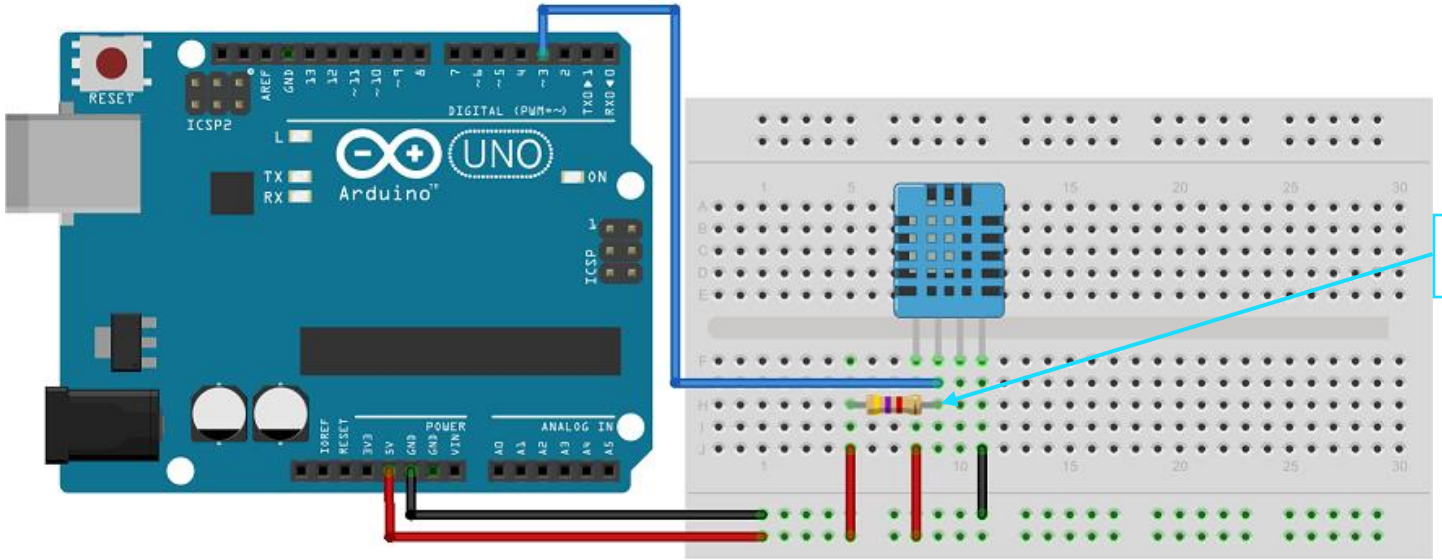
DHT11 (Humidity & Temperature Sensor)

- 결선

1)모듈형

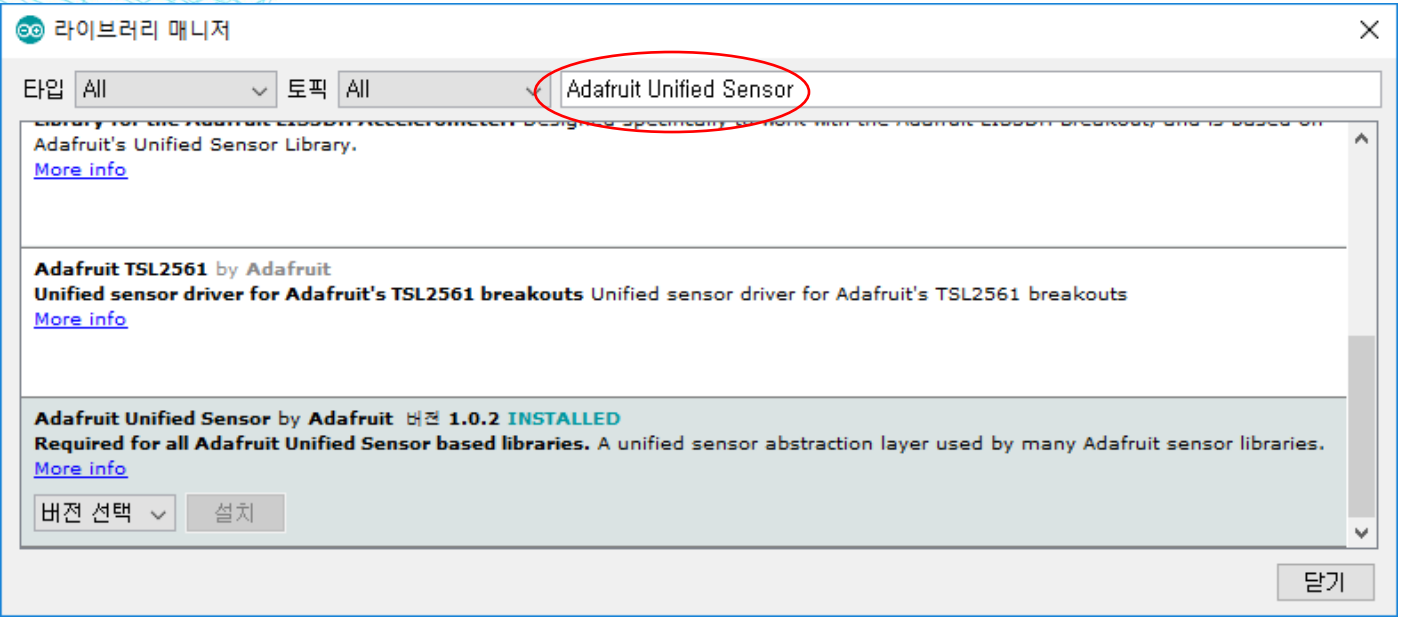


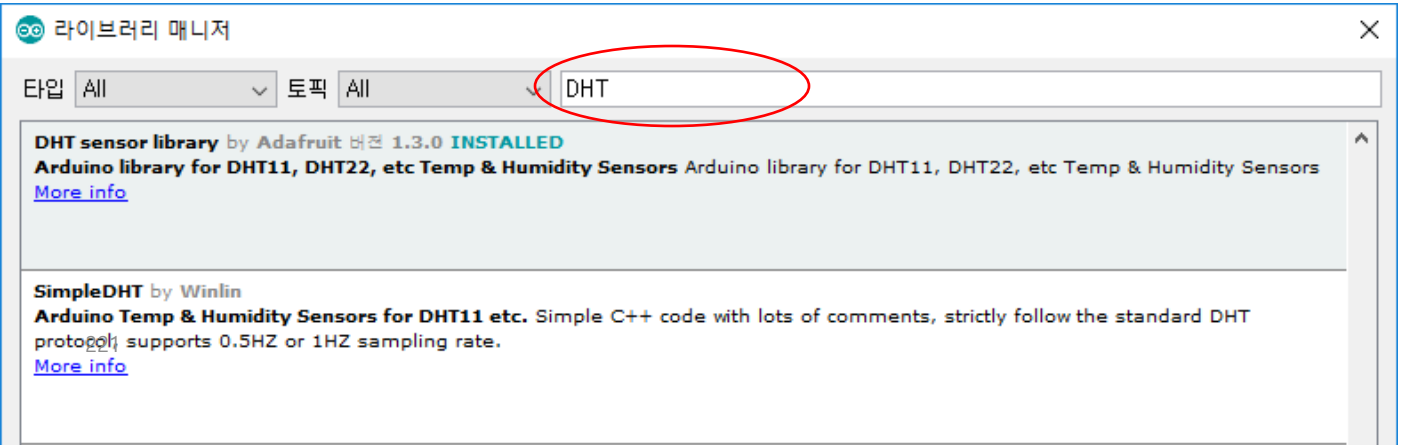
2)센서형



DHT11 (Humidity & Temperature Sensor)

- 라이브러리 설치





- 소스코드

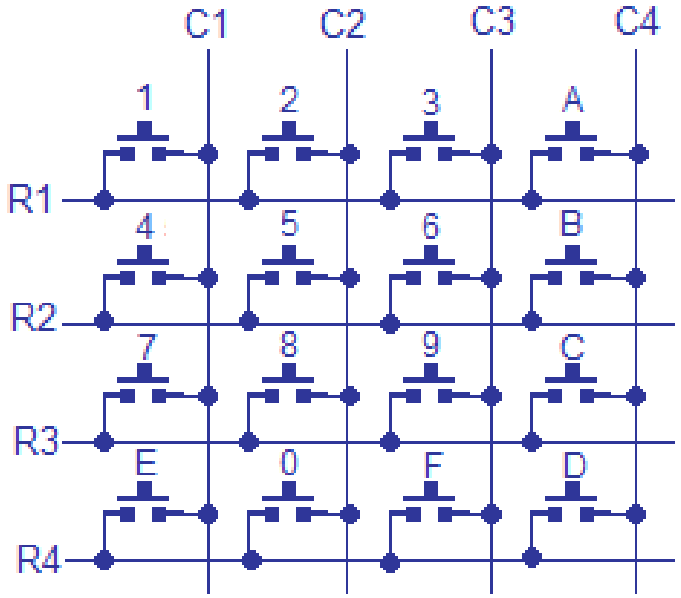
5-6-1.ino

```
#include "DHT.h"
#define DHTPIN 3
#define DHTTYPE DHT11

// DHT설정 - dht (디지털3, DHT11)
DHT dht(DHTPIN, DHTTYPE);

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

void loop() {
  delay(2000);
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  Serial.print("Humidity: ");
  Serial.print(h);
  Serial.print("%\t");
  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.println(" C");
}
```

4 x 4 Matrix Keypad

- 외관

- 내부 회로도



4 x 4 Matrix Keypad

패드	의미	Arduino
1	R1	**4**
2	R2	**5**
3	R3	**6**
4	R4	**7**
5	C1	**8**
6	C2	**9**
7	C3	**10**
8	C4	**11**

패드	의미	Arduino
R1	R1	**4**
R2	R2	**5**
R3	R3	**6**
R4	R4	**7**
C1	C1	**8**
C2	C2	**9**
C3	C3	**10**
C4	C4	**11**

```
/*4x4 Matrix Keypad connected to Arduino
This code prints the key pressed on the keypad to the serial port*/
#include <Keypad.h>
const byte numRows= 4; //number of rows on the keypad
const byte numCols= 4; //number of columns on the keypad

//keymap defines the key pressed according to the row and columns just as appears on the keypad
char keymap[numRows][numCols]= {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};

//Code that shows the the keypad connections to the arduino terminals
byte rowPins[numRows] = {4, 5, 6, 7};  //Rows 1 to 4
byte colPins[numCols] = {8, 9,10,11};  //Columns 1 to 4

//initializes an instance of the Keypad class
Keypad myKeypad= Keypad(makeKeymap(keymap), rowPins, colPins, numRows, numCols);

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

//If key is pressed, this key is stored in 'keypressed' variable
//If key is not equal to 'NO_KEY', then this key is printed out
void loop() {
  char keypressed = myKeypad.getKey();
  if (keypressed != NO_KEY) {
    Serial.print(keypressed);
  }
}
```

```
/*4x4 Matrix Keypad connected to Arduino
This code prints the key pressed on the keypad to the serial port*/
#include <Keypad.h>
const byte numRows= 4; //number of rows on the keypad
const byte numCols= 4; //number of columns on the keypad

//keymap defines the key pressed according to the row and columns just as appears on the keypad
char keymap[numRows][numCols]= {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};

//Code that shows the the keypad connections to the arduino terminals
byte rowPins[numRows] = {4, 5, 6, 7};  //Rows 1 to 4
byte colPins[numCols] = {8, 9,10,11};  //Columns 1 to 4

//initializes an instance of the Keypad class
Keypad myKeypad= Keypad(makeKeymap(keymap), rowPins, colPins, numRows, numCols);

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

//If key is pressed, this key is stored in 'keypressed' variable
//If key is not equal to 'NO_KEY', then this key is printed out
void loop() {
  char keypressed = myKeypad.getKey();
  if (keypressed != NO_KEY) {
    Serial.print(keypressed);
  }
}
```

난수 뽑기

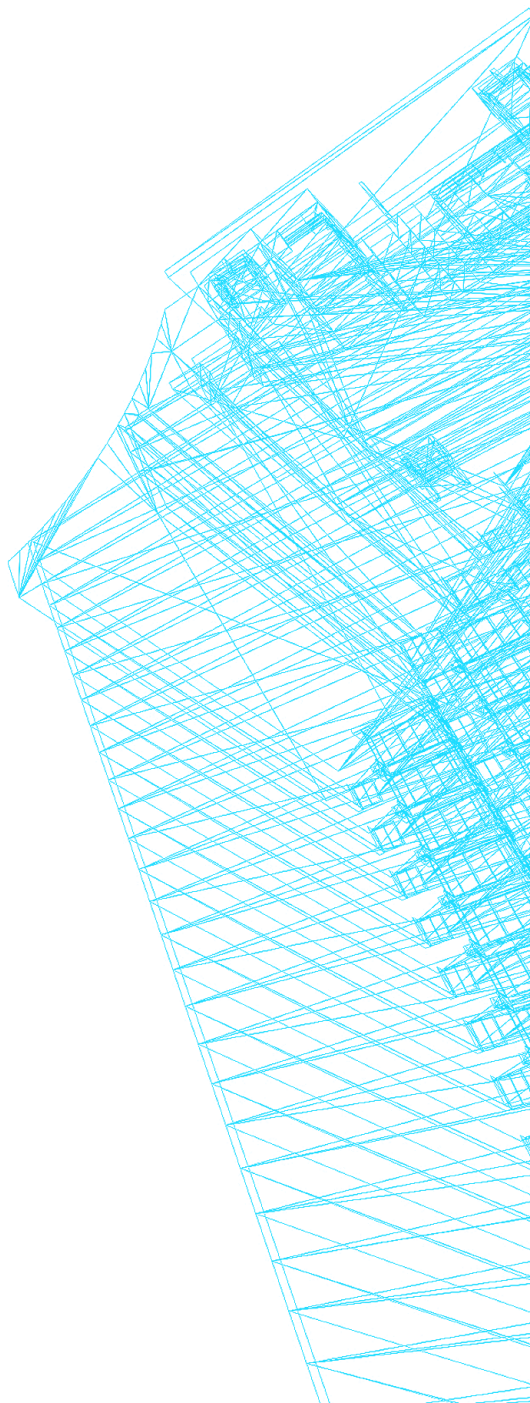
- Description
  - The random function generates pseudo-random numbers.
- Syntax
  - random(max)
  - random(min, max)
- Parameters
  - min: lower bound of the random value, inclusive (optional).
  - max: upper bound of the random value, exclusive.
- Returns
  - A random number between min and max-1.
  - Data type: long.

- Example coode

```
long randNumber;

void setup() {
  Serial.begin(9600);
  // if analog input pin 0 is unconnected, random analog
  // noise will cause the call to randomSeed() to generate
  // different seed numbers each time the sketch runs.
  // randomSeed() will then shuffle the random function.
  randomSeed(analogRead(0));
}

void loop() {
  // print a random number from 0 to 299
  randNumber = random(300);
  Serial.println(randNumber);

  // print a random number from 10 to 19
  randNumber = random(10, 20);
  Serial.println(randNumber);
  delay(50);
}
```

Motor

- DC Motor
- Servo Motor
- Stepping Motor

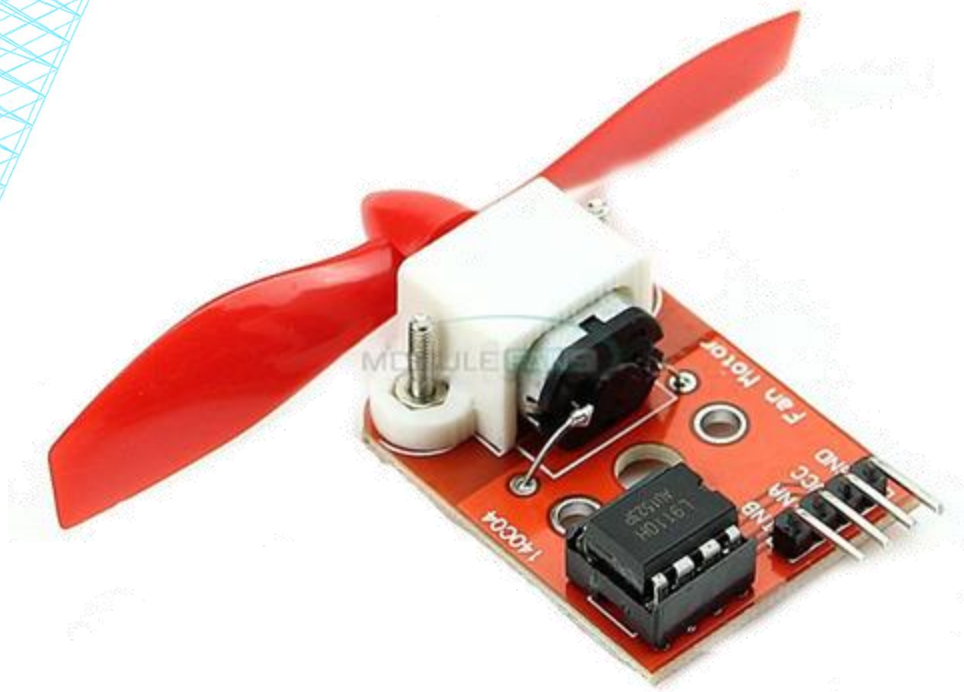


Motor의 종류와 특징

- **DC 모터**
  - **주변에서 흔히 보는 모터**. 입력 전류(+, -) 방향으로 회전 방향을 제어할 수 있음. 상대적으로 고회전에 유리. 회전 움직임을 사용하는 RC카, 쿼드콥터 등 사용처가 매우 다양. **회전수와 방향을 자유롭게 제어하기 위해서는 별도의 드라이버 모듈이 필요**.
- **서보 모터**
  - **보통 0~180도 사이를 움직이며**, 해당 회전 범위 안에서의 위치를 사용자가 설정 가능. 동작 범위가 제한적이지만 정확한 위치 제어가 가능. 제어 방법도 간단함. (PWM 신호로 간단히 위치제어) RC카의 방향타, 로봇 관절 등 회전각 제어가 필요한 곳에 광범위하게 사용.
- **스텝 모터**
  - 회전 방향과 속도 뿐 아니라 회전각을 정밀히 제어할 수 있음. DC 모터와 서보 모터의 장점을 합친 모터. 하지만 제어가 복잡함. 그래서 보통 스텝모터 드라이버 모듈을 이용해서 제어. 상대적으로 고회전이 필요치 않으면서 정밀한 제어가 필요한 곳에 사용. 3D 프린터 움직임을 만드는 핵심 모터이기도 함.

DC모터

- 외형



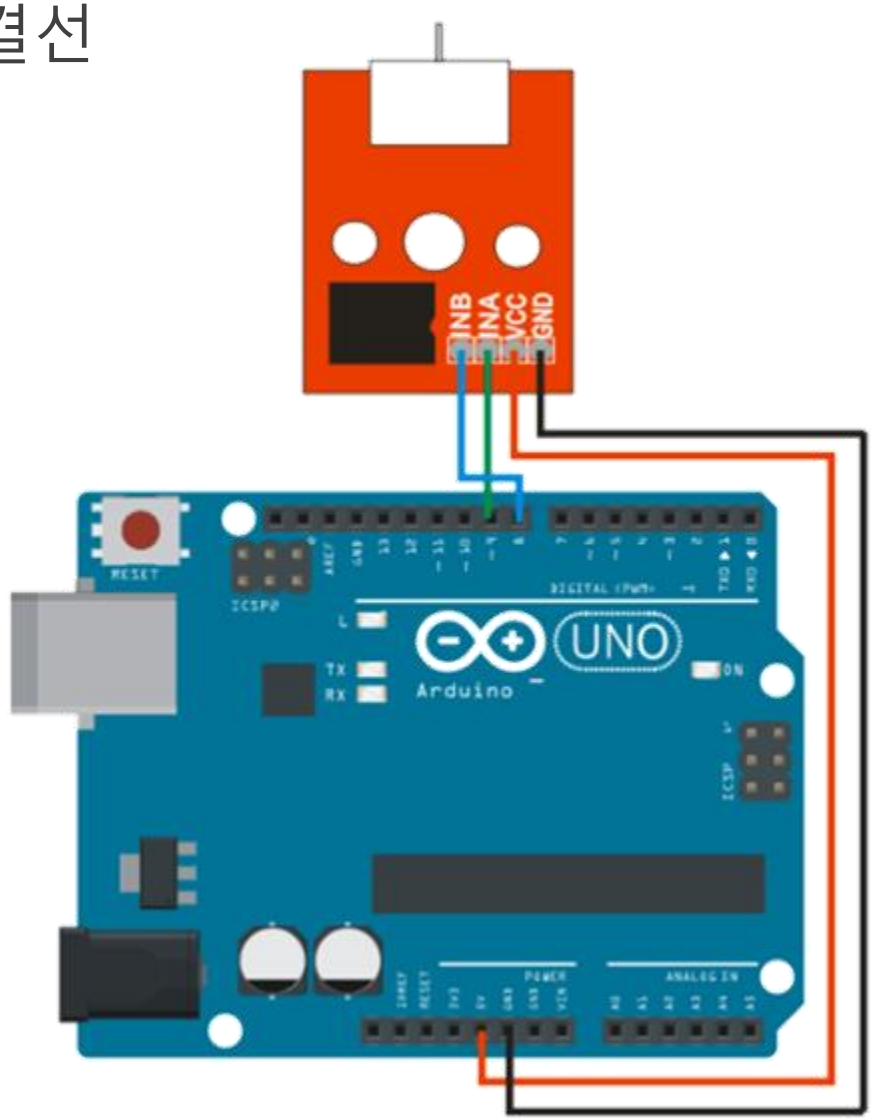
- 특징
  - L9110 drive, speed control with ZhuanPei mounting holes, screw compatible steering gear steering wheel control high quality, high efficiency.
  - Can easily blow out 20 cm away lighter flame can be used for fire fighting robots, robot for design and development
  - Working voltage: 5v
  - Board size: (L*W*H)3.49*2.15*0.15cm/1.37"*0.85"*0.06"(Approx.)
  - Propeller diameter: 7.5cm/2.95"(Approx.) .

DC모터

- L9110 모터 드라이버

INA	INB	움직임
0	0	정지
0	1	시계 방향 회전
1	1	정지
1	0	반시계 방향 회전

- 결선



DC모터

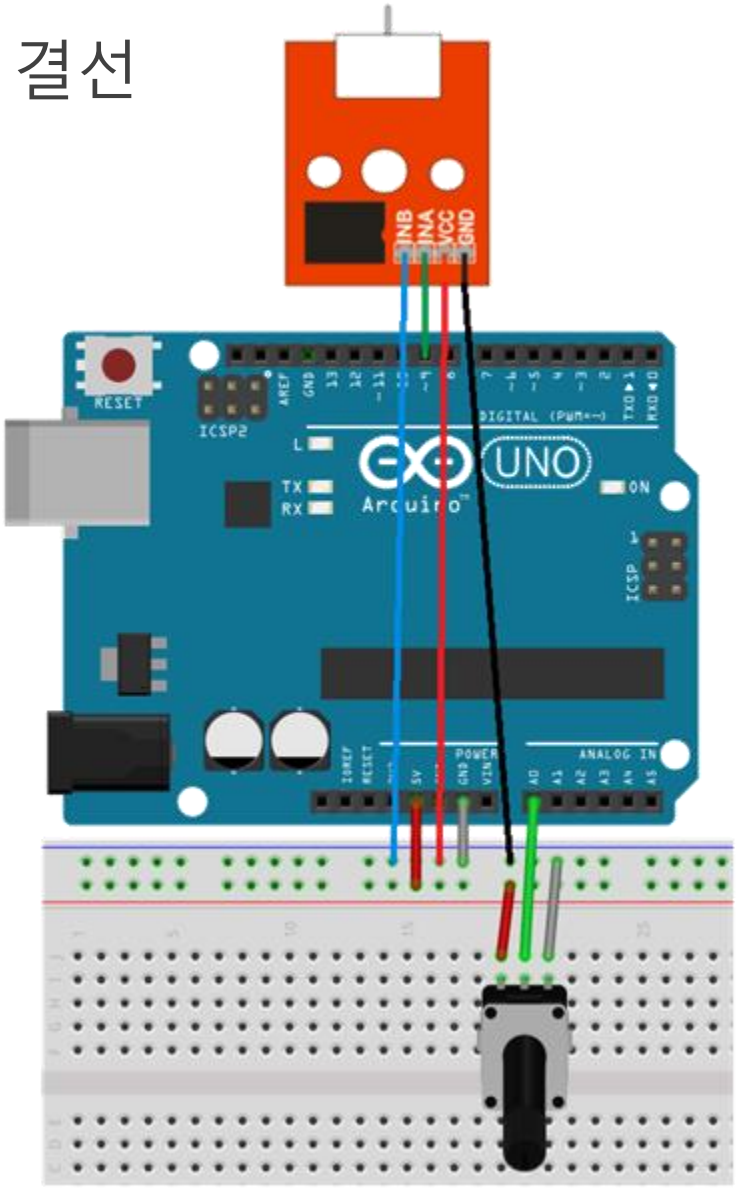
5-1-1.ino

```
#define INA  9
#define INB  8

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

int a=0, b=0;
int flag=0;
unsigned long past=0;
```

```
void loop() {
  unsigned long current = millis();
  if(current-past > 3000) {
    switch(flag%4) {
      case 0 : a = 0; b = 0;  break;
      case 1 : a = 1; b = 0;  break;
      case 2 : a = 1; b = 1;  break;
      case 3 : a = 0; b = 1;  break;
    }
    digitalWrite(INA, a);
    digitalWrite(INB, b);

    Serial.print("a:b = ");
    Serial.print(a);
    Serial.print(":");
    Serial.println(b);
    past = current;
    flag++;
  }
}
```

DC모터 (속도제어)

- 결선

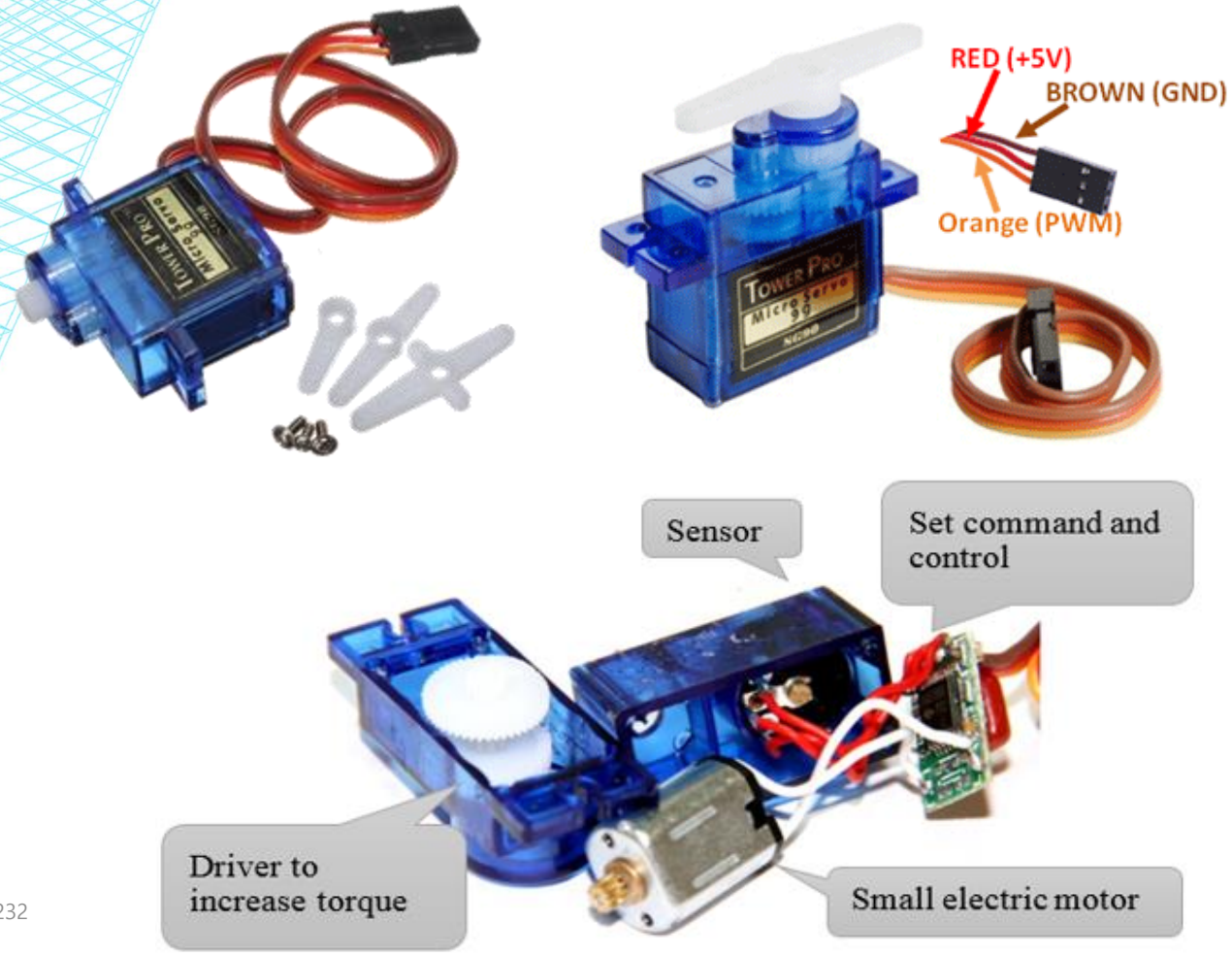


5-1-2.ino

```
#define INA  9
#define POT  A0

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

void loop() {
  int p = analogRead(POT);
  int v = map(p, 0, 1023, 0, 255);
  Serial.print("Speed : ");
  Serial.println(v);
  analogWrite(INA, v);
  delay(500);
}
```

서보모터 (SG90)

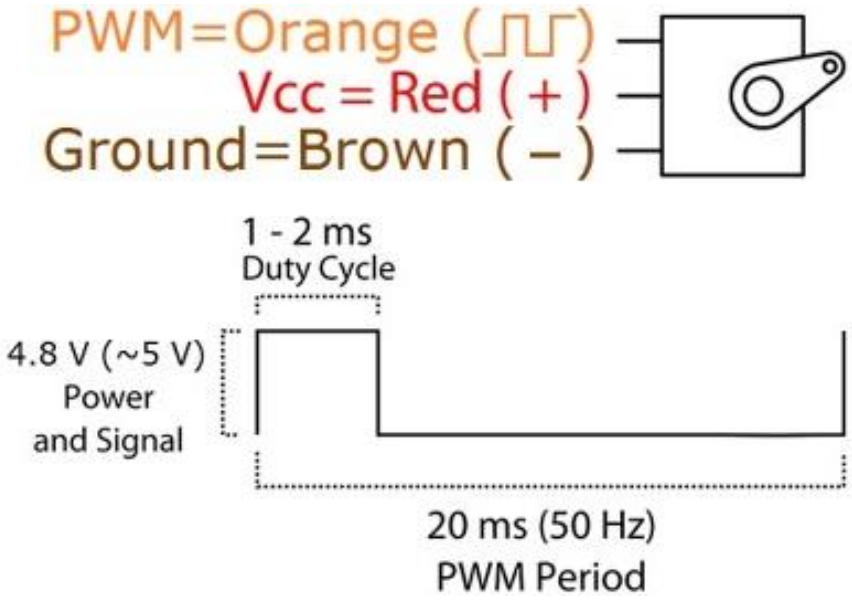
- 외형



- 특징
  - 아두이노의 제한된 전원으로도 충분히 동작-제어가 가능한 **마이크로 서보모터(SG-90)**
  - 비록 힘은 딸리지만 작고, 값도 싸고, 아두이노의 전원(5V 핀)으로도 동작합니다. 더 큰 힘(토크)이 필요하다면 MG-9xx 시리즈와 같은 고출력 모델을 구입해서 사용하면 됩니다만 이 경우는 별도의 외부 전원에서 모터에 전력을 공급해 줘야함.

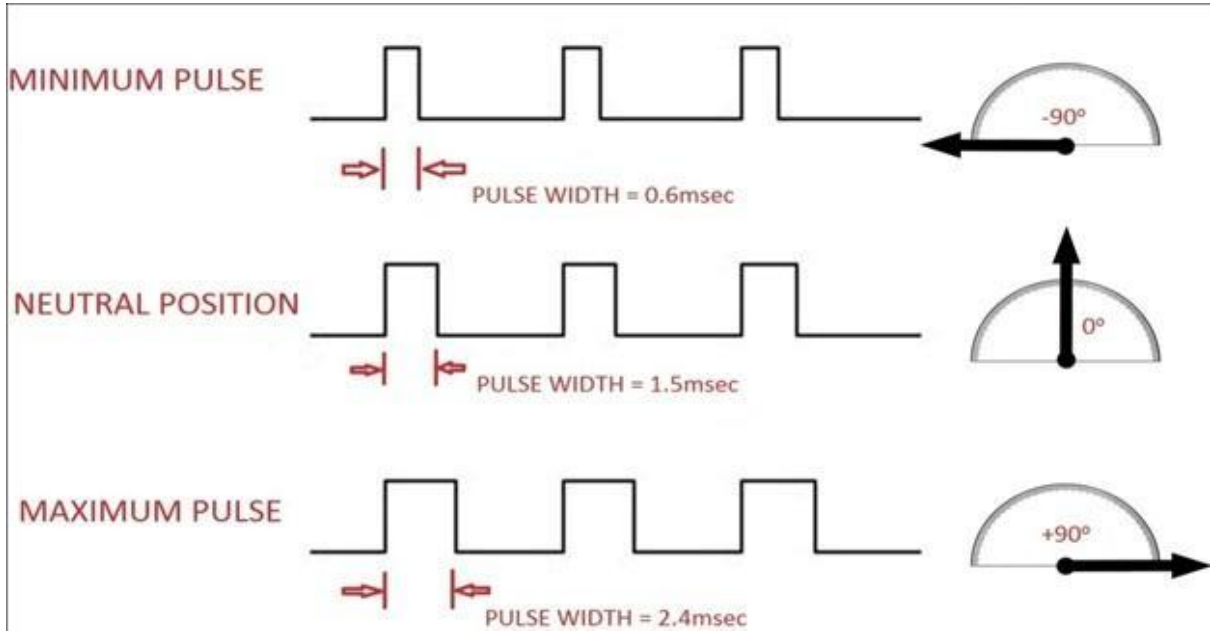
서보모터 (SG90)

- 연결



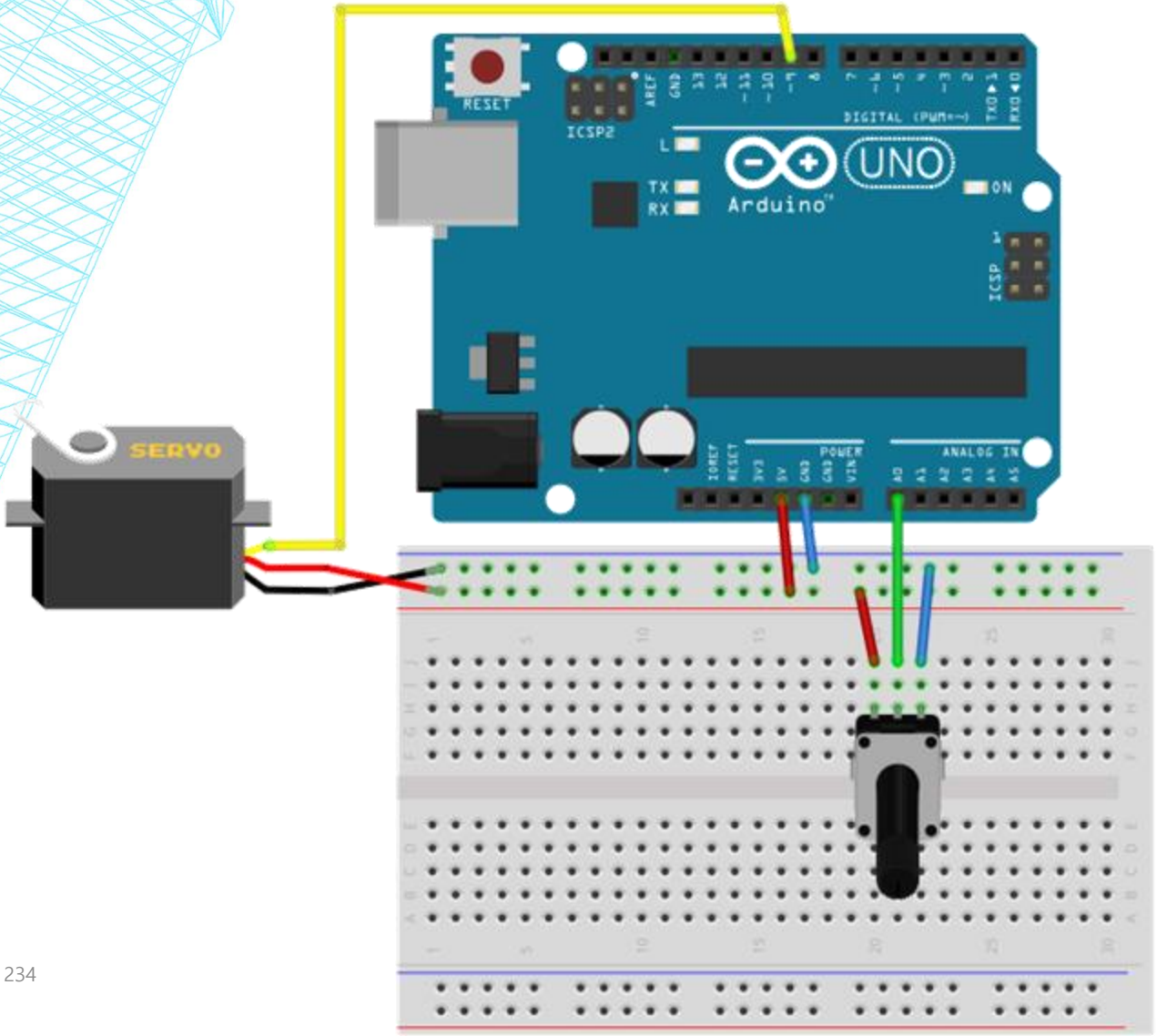
서보모터	아두이노
검정색 혹은 갈색	GND
적색	5V
황색	D9

- 제어



- Position "0" (1.5ms pulse) is middle, "90" (~2ms pulse) is all the way to the right, "-90" (~1ms pulse) is allthe way to the left.

서보모터 (SG90)



5-2-1.ino

```
#include <Servo.h>
#define SERVO 9
#define POT  A0

Servo servo;

void setup() {
  servo.attach(SERVO);
  servo.write(0);          //   0도 = 왼쪽 회전 한계
  delay(2000);
  servo.write(180);        // 180도 = 오른쪽 회전 한계
  delay(2000);
  servo.write(90);         //  90도 = 중앙
  delay(2000);
}

void loop() {
  int potVal = analogRead(POT);
  int angle = map(potVal, 0, 1023, 0, 180);
  servo.write(angle);
  delay(100);
}
```

서보모터 활용 프로젝트 예

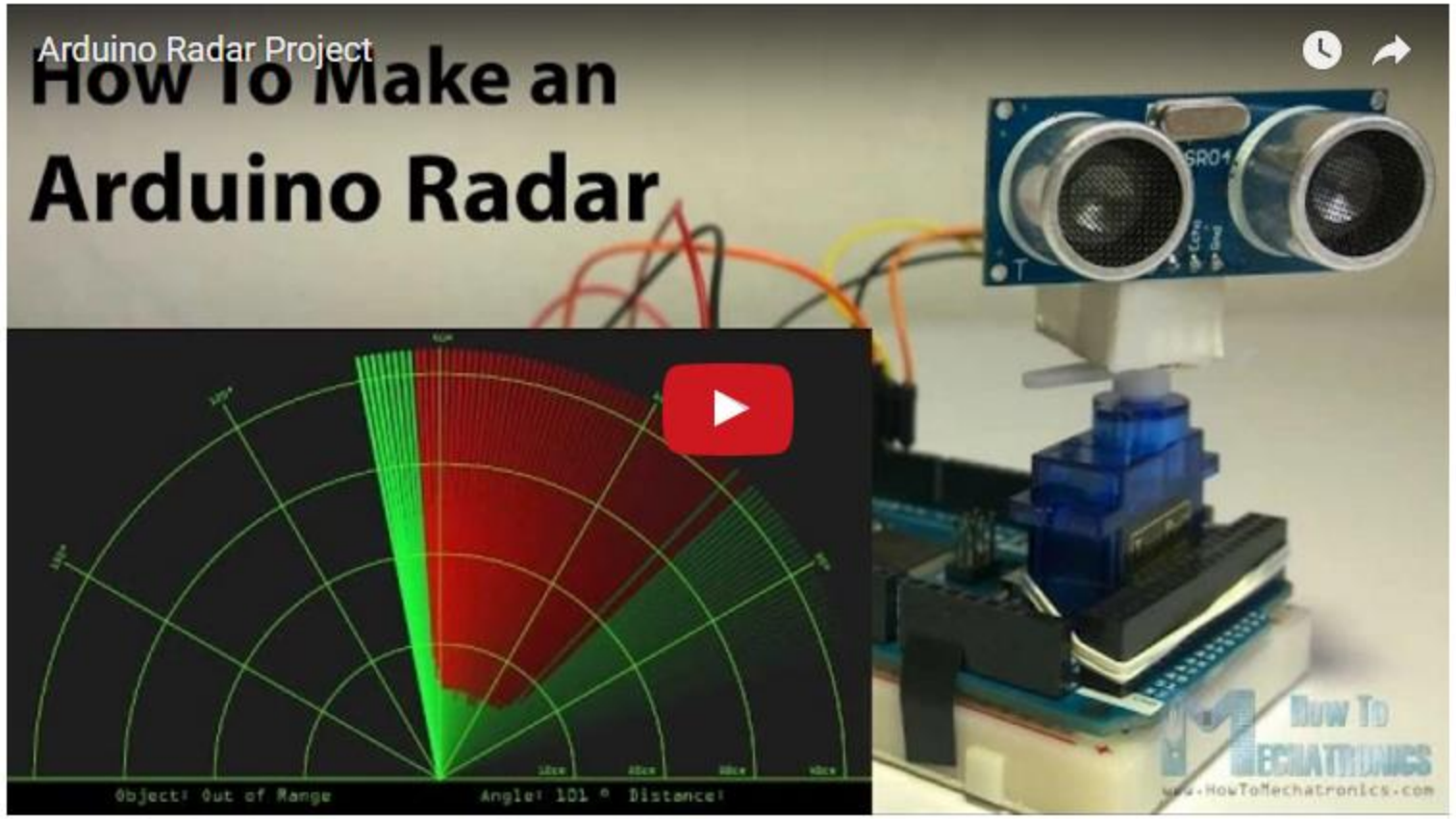
초음파센서 + 서보모터 응용

Arduino Radar Project

http://howtomechatronics.com/projects/arduino-radar-project/

Dejan Nedelkovski July 28, 2015 Projects 292

In this Arduino Tutorial I will show you how you can make this cool looking radar using the Arduino Board and the Processing Development Environment. You can watch the following video or read the written tutorial below for more details.




END