

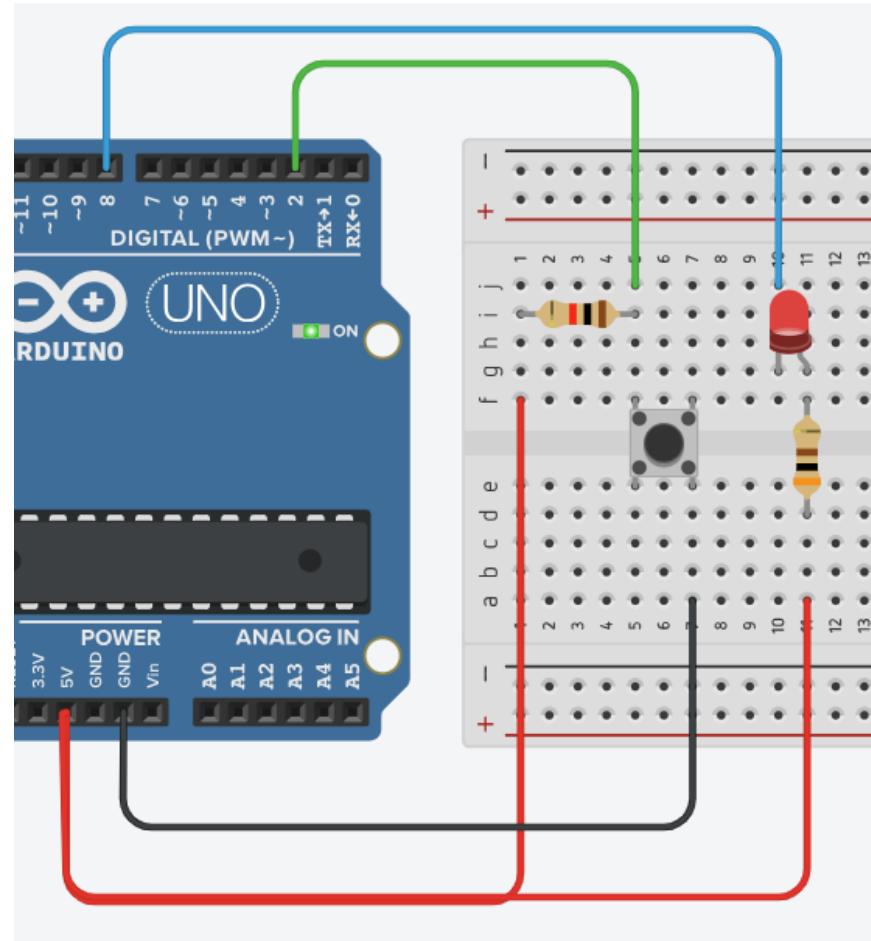
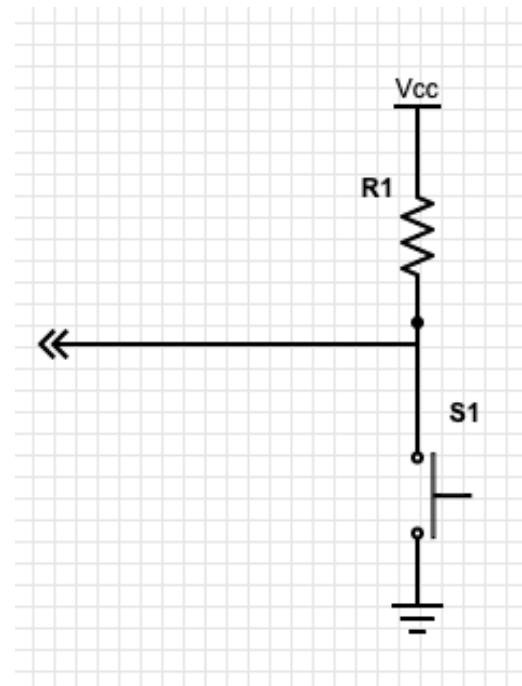
# 인터럽트 기반의 디지털 센서 실험

# 목표

- 인터럽트를 이용한 디지털(ON/OFF) 센서 실험
- 인터럽트를 이용한 화재 감지 실험
- 비접촉식 온도 센서 모듈 실험

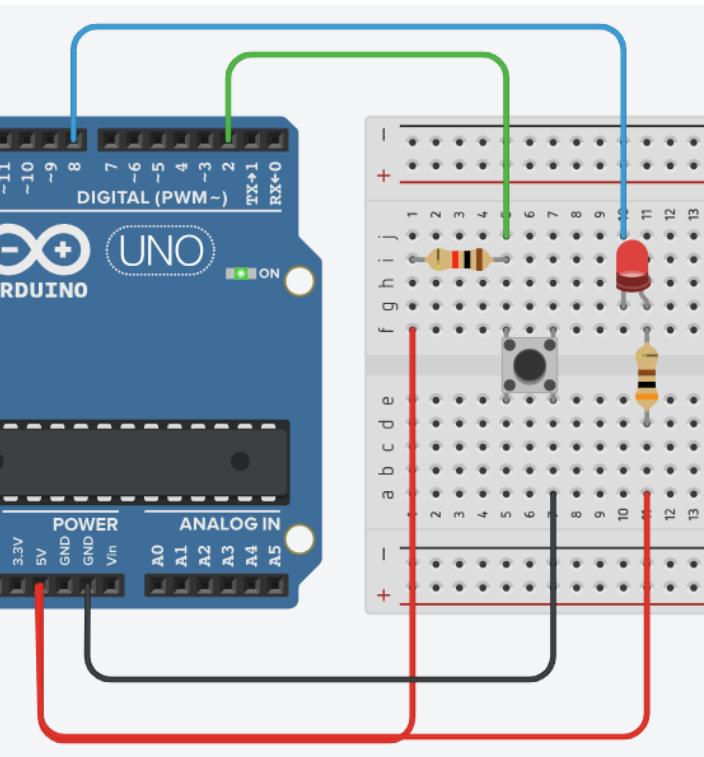
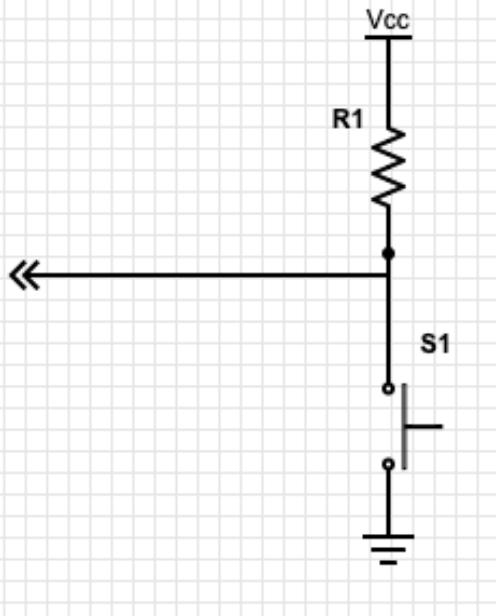
# 외부 인터럽트(External Interrupt)

- 스위치 vs 인터럽트



# 외부 인터럽트(External Interrupt)

- 스위치 VS 인터럽트



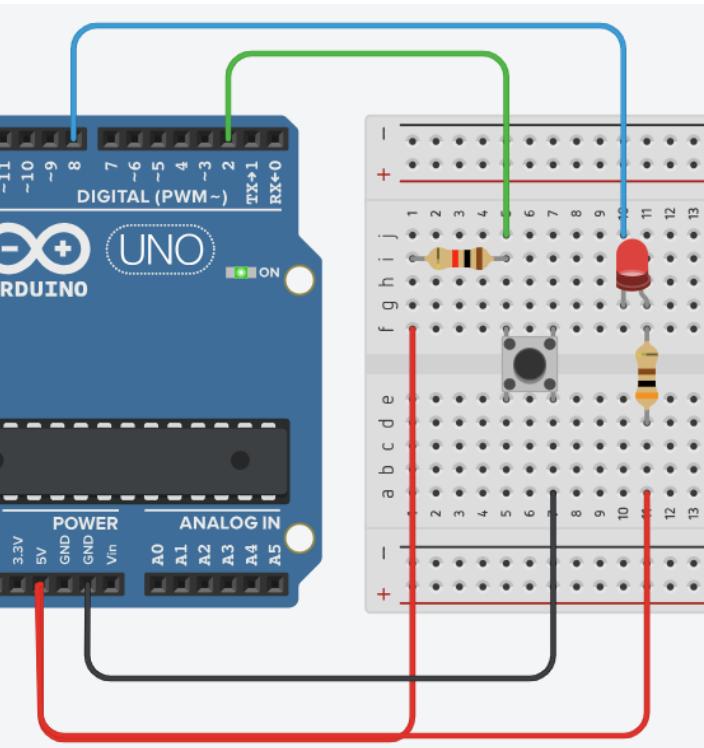
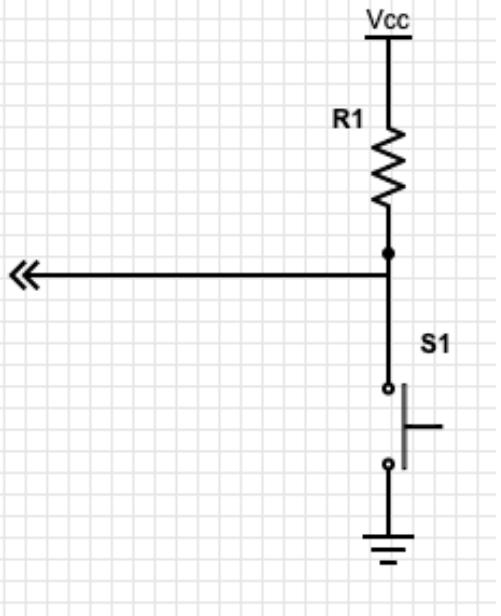
```
void setup()
{
    pinMode(2, INPUT);
    pinMode(8, OUTPUT);
}

void loop()
{
    int input = digitalRead(2);

    if( input == 0 )
    {
        digitalWrite(8, 0);
    }
    else
    {
        digitalWrite(8, 1);
    }
}
```

# 외부 인터럽트(External Interrupt)

- 스위치 VS 인터럽트



```
void setup()
{
    pinMode(2, INPUT);
    pinMode(8, OUTPUT);

    Serial.begin(9600);
}

void loop()
{

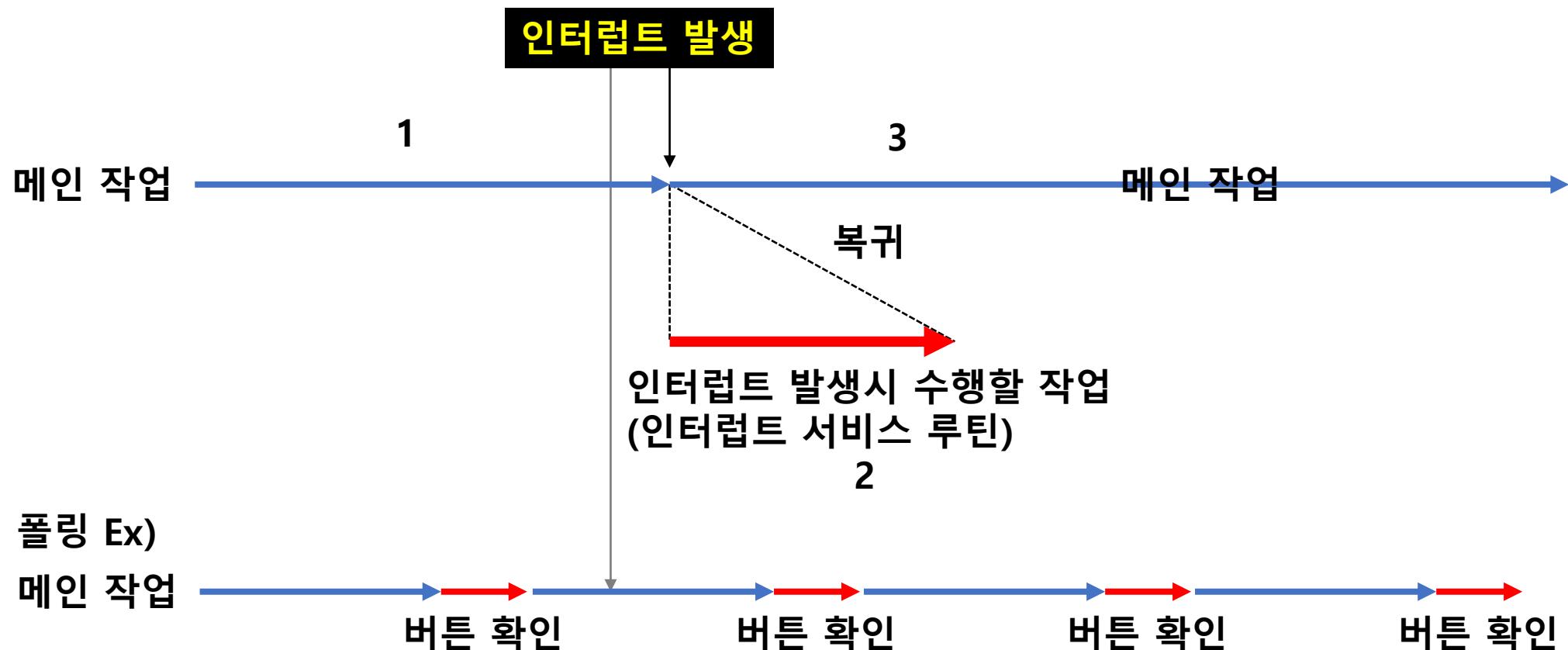
    digitalWrite(8, 0);
    delay(1000);

    digitalWrite(8, 1);
    delay(1000);

    int input = digitalRead(2);
    if( input == 0 )
    {
        Serial.println("key");
    }
}
```

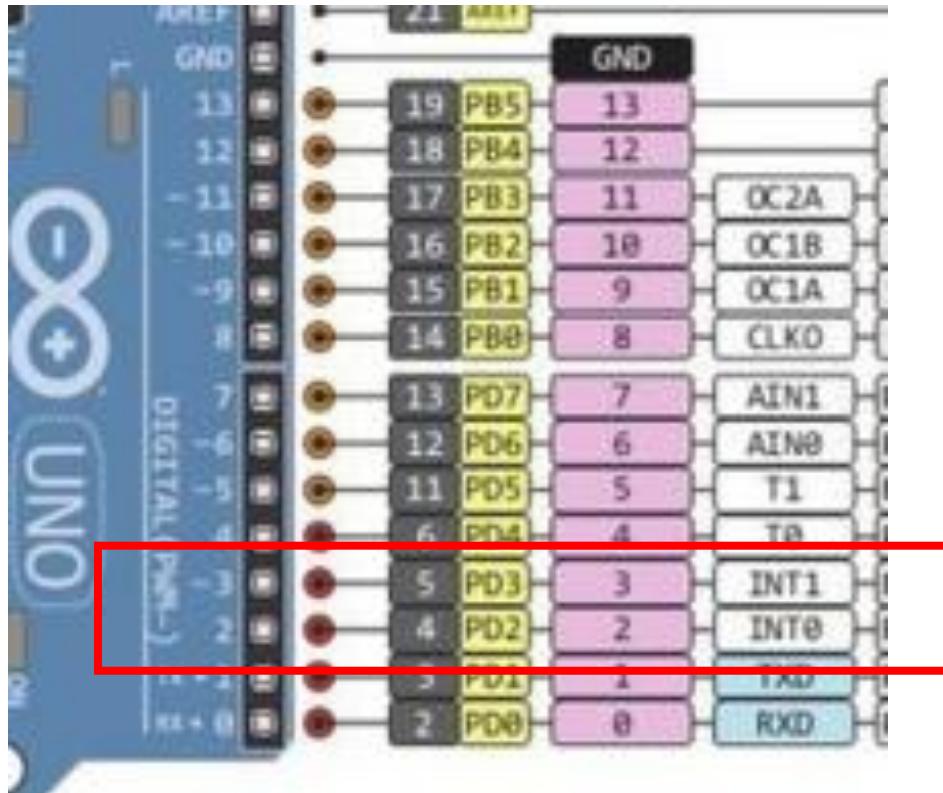
# 외부 인터럽트(External Interrupt)

- 폴링 vs 인터럽트



# 외부 인터럽트(External Interrupt)

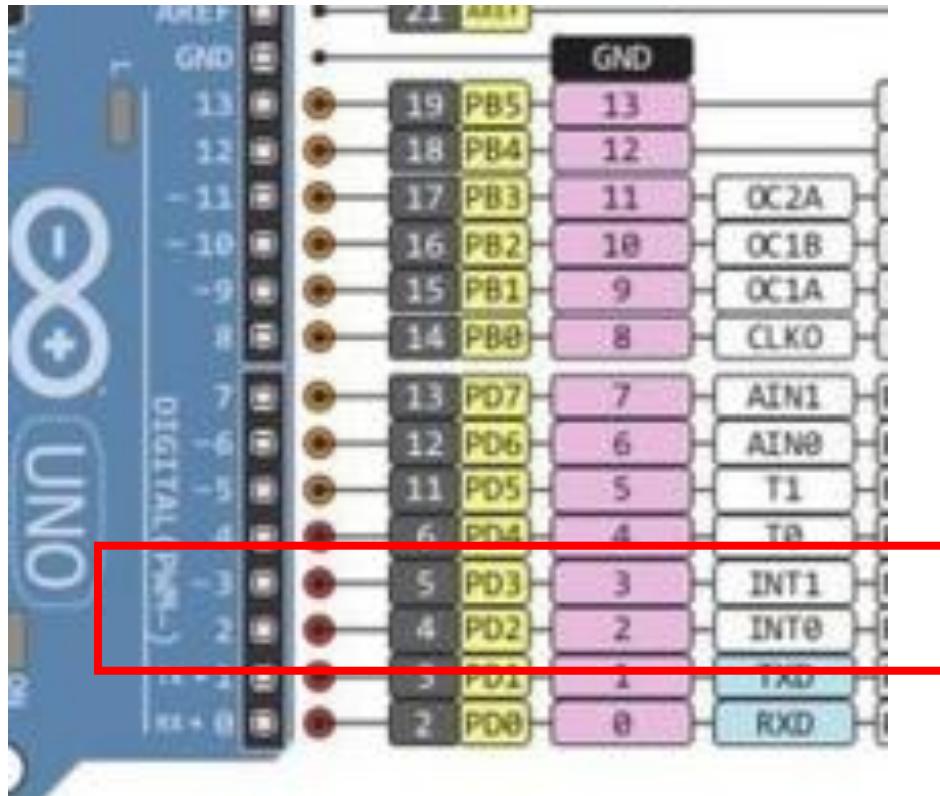
- 폴링 vs 인터럽트



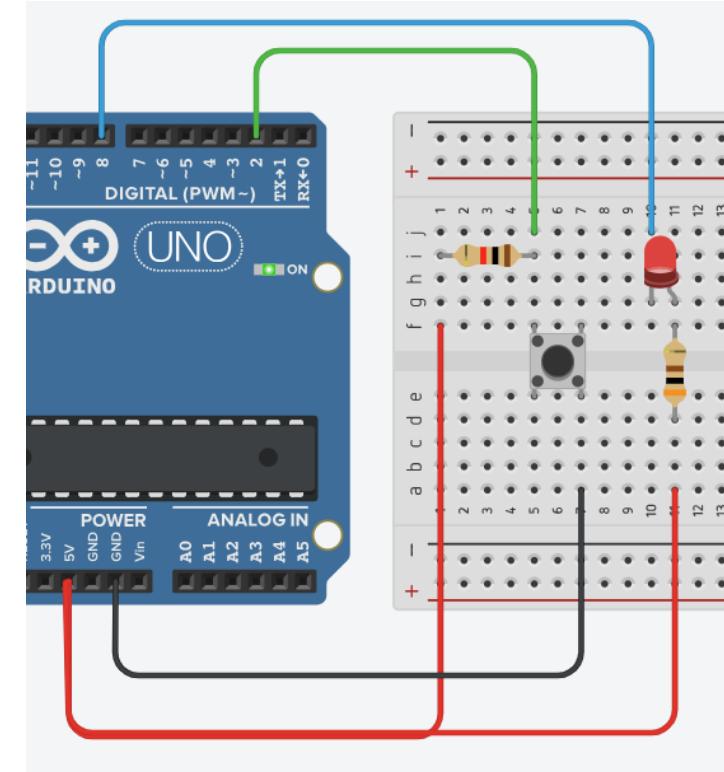
- INT1 : Interrupt #1
- INT0 : Interrupt #0

# 외부 인터럽트(External Interrupt)

- 빠른 vs 인터럽트

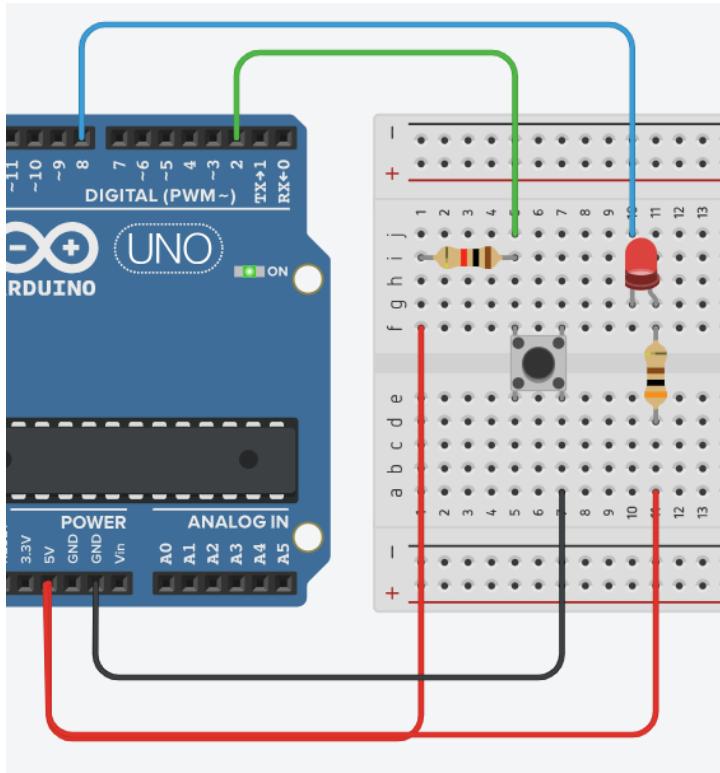


$2 \rightarrow \text{INT0} : \text{Interrupt } \#0$



# 외부 인터럽트(External Interrupt)

- 폴링 vs 인터럽트
- 2 → INT0 : Interrupt #0

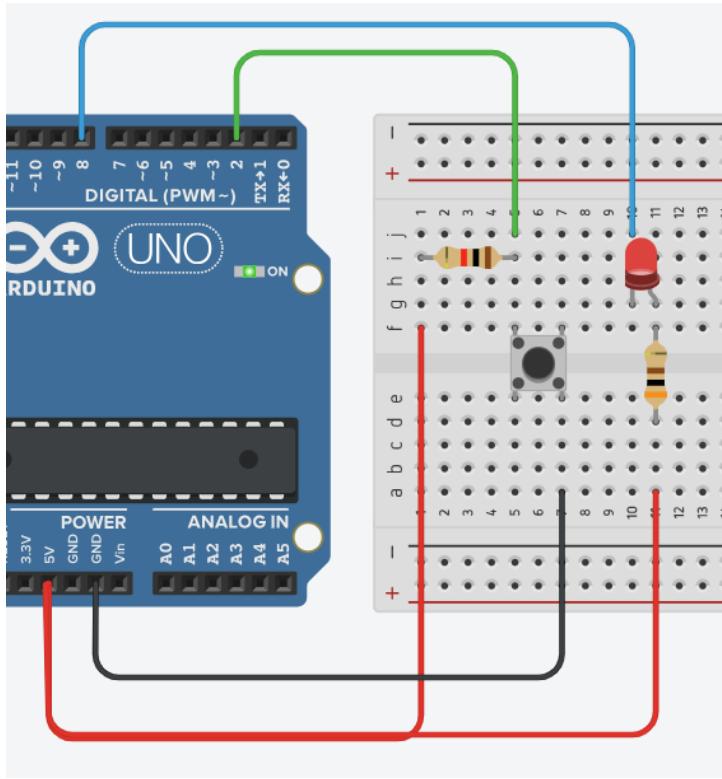


인터럽트 발동 조건 (mode)

모드	상태
LOW	핀이 LOW일때
CHANGE	LOW->HIGH or HIGH->LOW로 변할 때
RISING	LOW ->HIGH일때
FALLING	HIGH -> LOW일때
HIGH	핀이 HIGH일때

# 외부 인터럽트(External Interrupt)

- 폴링 vs 인터럽트
- 2 → INT0 : Interrupt #0



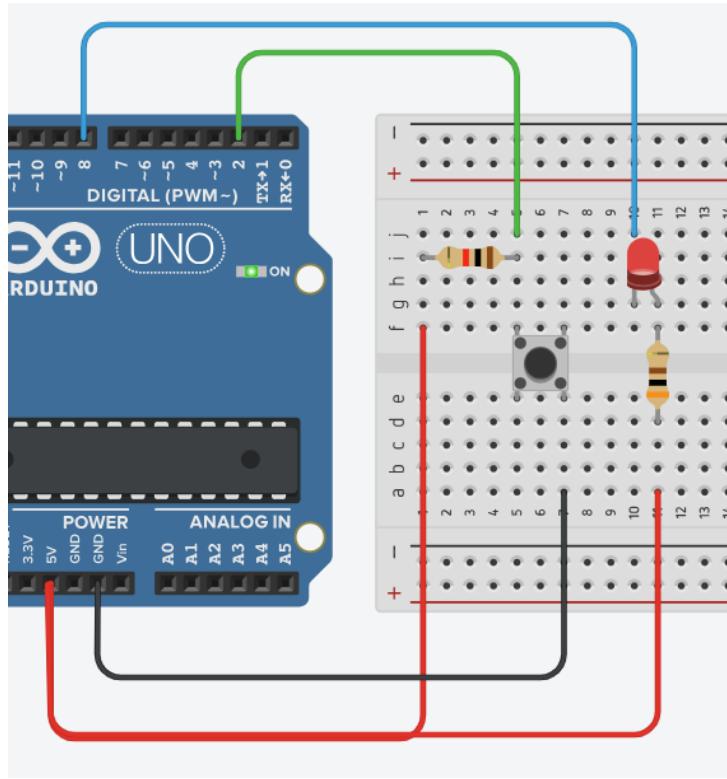
```
attachInterrupt( digitalPinToInterrupt(핀번호), 서비스루틴함수명, 모드 );
```

## 인터럽트 발동 조건 (mode)

모드	상태
LOW	핀이 LOW일때
CHANGE	LOW->HIGH or HIGH->LOW로 변할 때
RISING	LOW ->HIGH일때
FALLING	HIGH -> LOW일때
HIGH	핀이 HIGH일때

# 외부 인터럽트(External Interrupt)

- 폴링 vs 인터럽트
- 2 → INT0 : Interrupt #0



`attachInterrupt( digitalPinToInterrupt(2), ExINT, FALLING );`

`attachInterrupt( digitalPinToInterrupt(핀번호), 서비스루틴함수명, 모드 );`

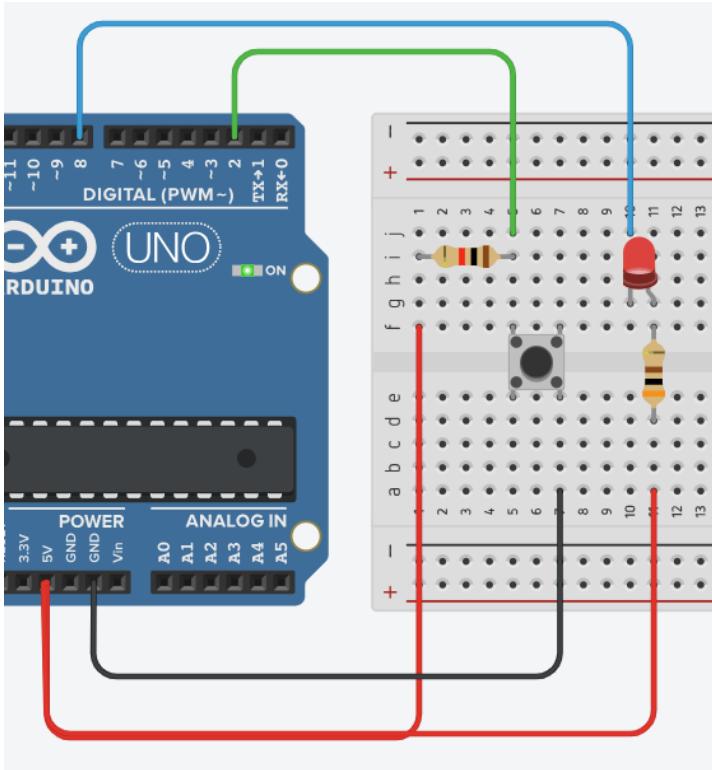
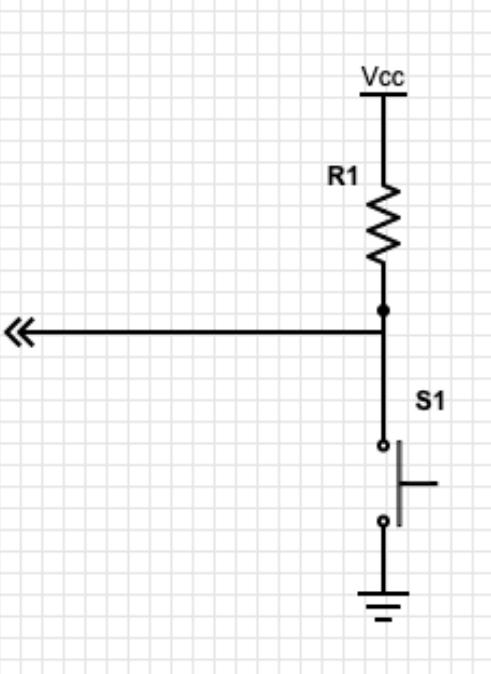
## 인터럽트 발동 조건 (mode)

모드	상태
LOW	핀이 LOW일때
CHANGE	LOW->HIGH or HIGH->LOW로 변할 때
RISING	LOW ->HIGH일때
FALLING	HIGH -> LOW일때
HIGH	핀이 HIGH일때

# 외부 인터럽트(External Interrupt)

- ▶ 블링 vs 인터럽트

`attachInterrupt( digitalPinToInterrupt(2), ExINT, FALLING );`



```
void setup()
{
    pinMode(8, INPUT);
    pinMode(2, OUTPUT);

    attachInterrupt( digitalPinToInterrupt(2), ExINT, FALLING );

    Serial.begin(9600);
}

void loop()
{
    digitalWrite(8, 0);
    delay(1000);

    digitalWrite(8, 1);
    delay(1000);
}

void ExINT()
{
    Serial.println("ExINT");
}
```

# 부저 실험

## • 부저(소리) 출력 실험

- 능동부저: 전원을 공급하면 단음(삐) 소리가 출력
- 수동부저: 진동을 만들어 특정 주파수의 소리를 출력(다양한 소리를 출력할 수 있음, 멜로디)

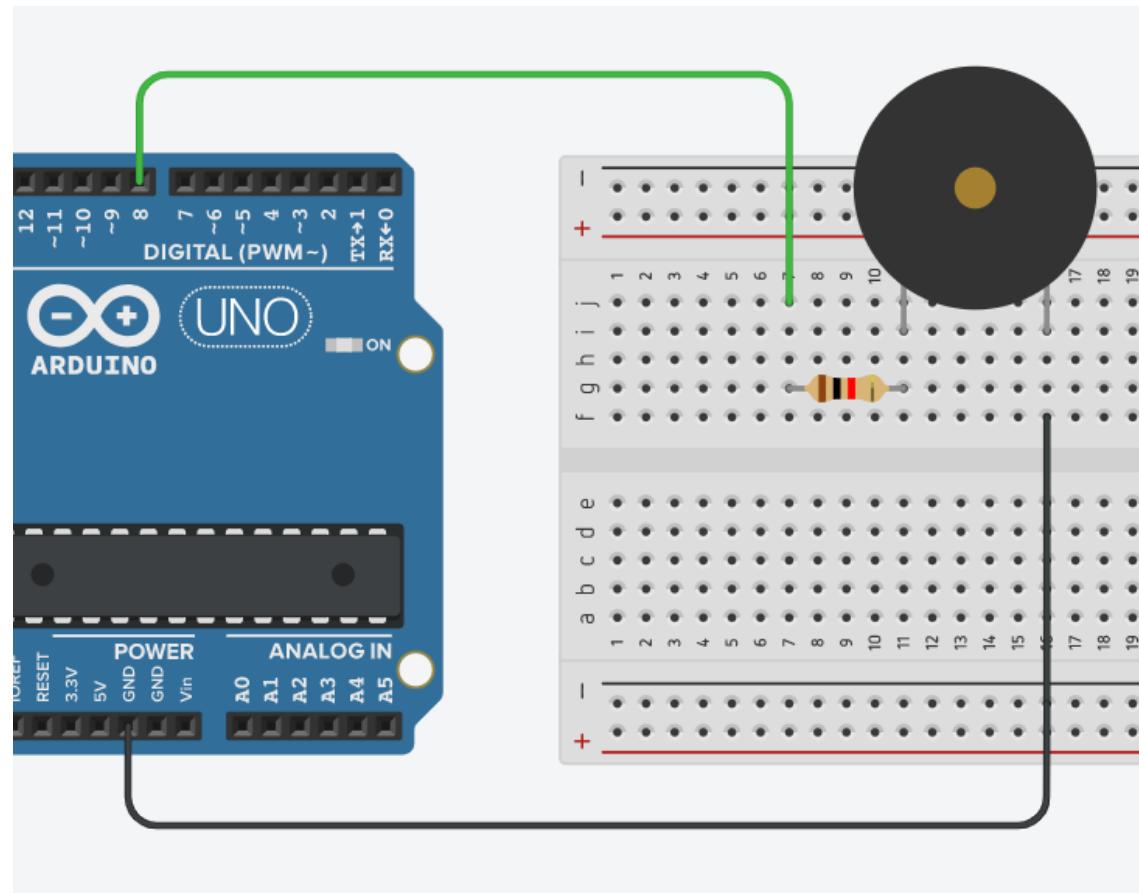


음계	1	2	3	4	5	6	7	8
C(도)	32.7032	65.4064	130.8128	261.6256	523.2511	1046.502	2093.005	4186.009
C#	34.6478	69.2957	138.5913	277.1826	554.3653	1108.731	2217.461	4434.922
D(레)	36.7081	73.4162	146.8324	293.6648	587.3295	1174.659	2349.318	4698.636
D#	38.8909	77.7817	155.5635	311.1270	622.2540	1244.508	2489.016	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

- 도: 261.6256 Hz
- 레: 293.1826 Hz
- 미: 329.6276 Hz
- 파: 349.2282 Hz
- 솔: 391.9954 Hz
- 라: 440.0000 Hz
- 시: 466.1638 Hz
- 도: 523.2511 Hz

# 부저 실험

- 부저(소리) 출력 실험
  - 부저 + <> 아두이노 8번핀
  - 부저 - <> 아두이노 GND



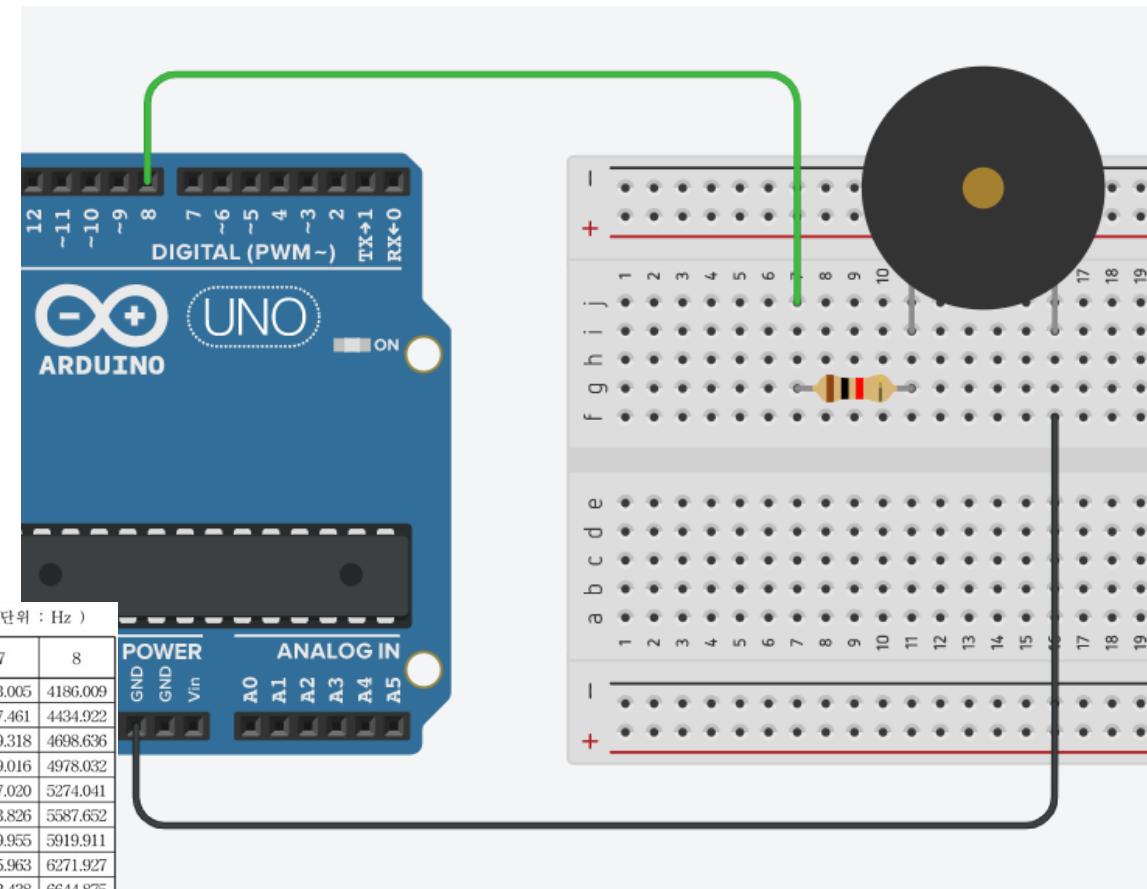
# 부저 + LED 실험

- 부저(소리) 출력 실험

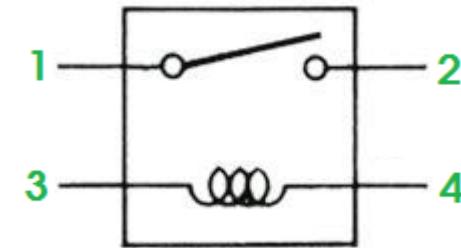
```
void setup()
{
    pinMode(8, OUTPUT);
}

void loop()
{
    tone(8, 262, 500);
    delay(500);
}
```

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



# 마그네틱 도어센서 실험



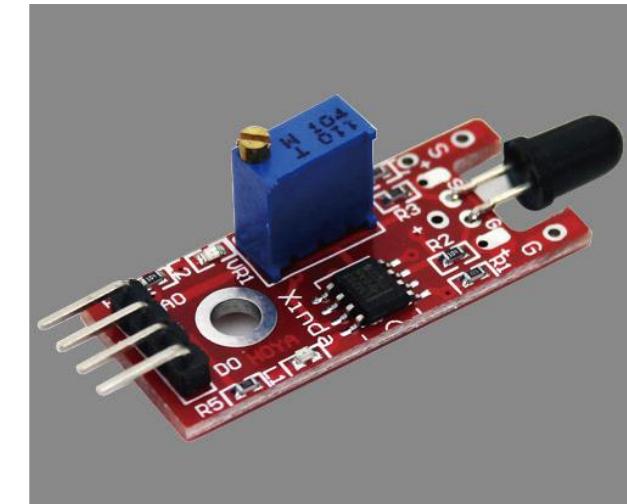
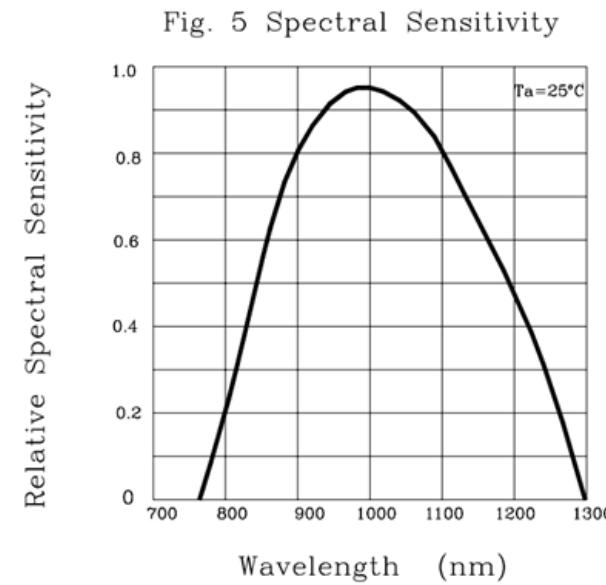
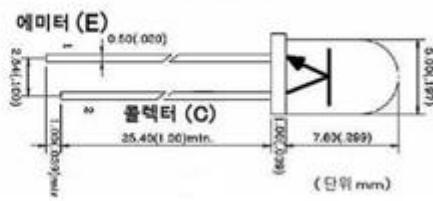
# QUIZ



문이 열리면(버튼이 눌렸을 때) 경고음(부저)을 울려봅시다.

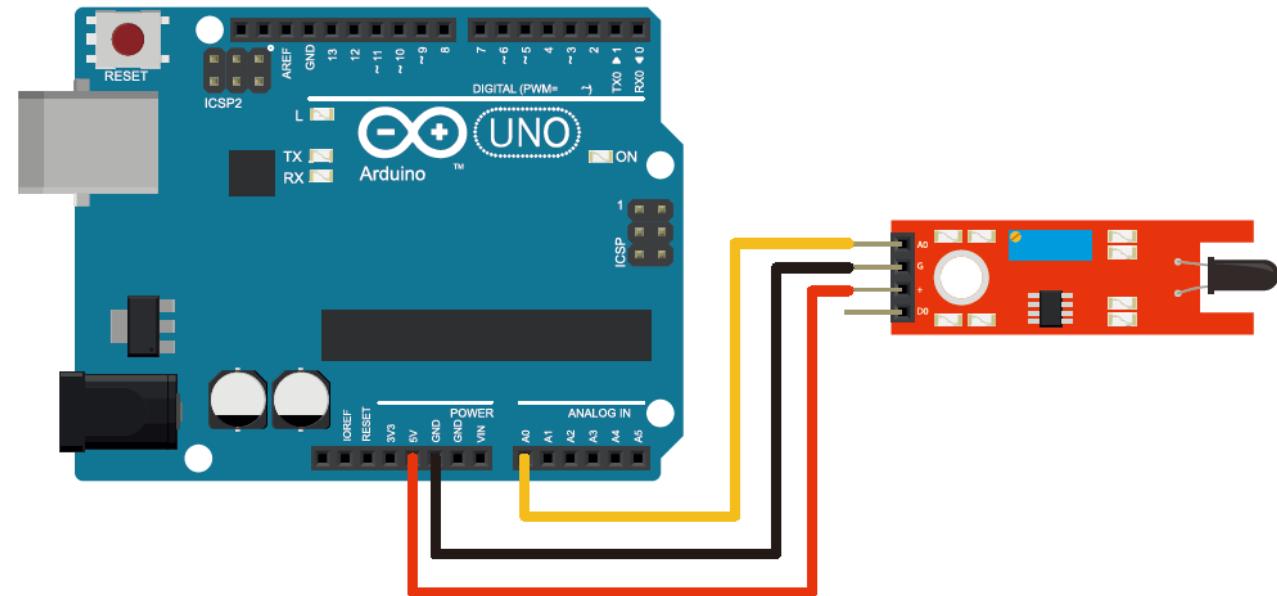
# 불꽃감지센서 (Flame sensor)

- 불꽃 또는 화염은 사람의 눈으로 확인 할 수 없는 자외선과 적외선의 파장이 발생
- 불꽃감지센서는 적외선 감지센서로서 760nm ~ 1100nm파장을 감지한다.



# 불꽃감지센서 (Flame sensor)

- 불꽃 감지 아두이노 실험 구성
  - 센서모듈 A0 <> 아두이노 A0
  - 센서모듈 G <> 아두이노 GND
  - 센서모듈 + <> 아두이노 5V



# 불꽃감지센서 (Flame sensor)

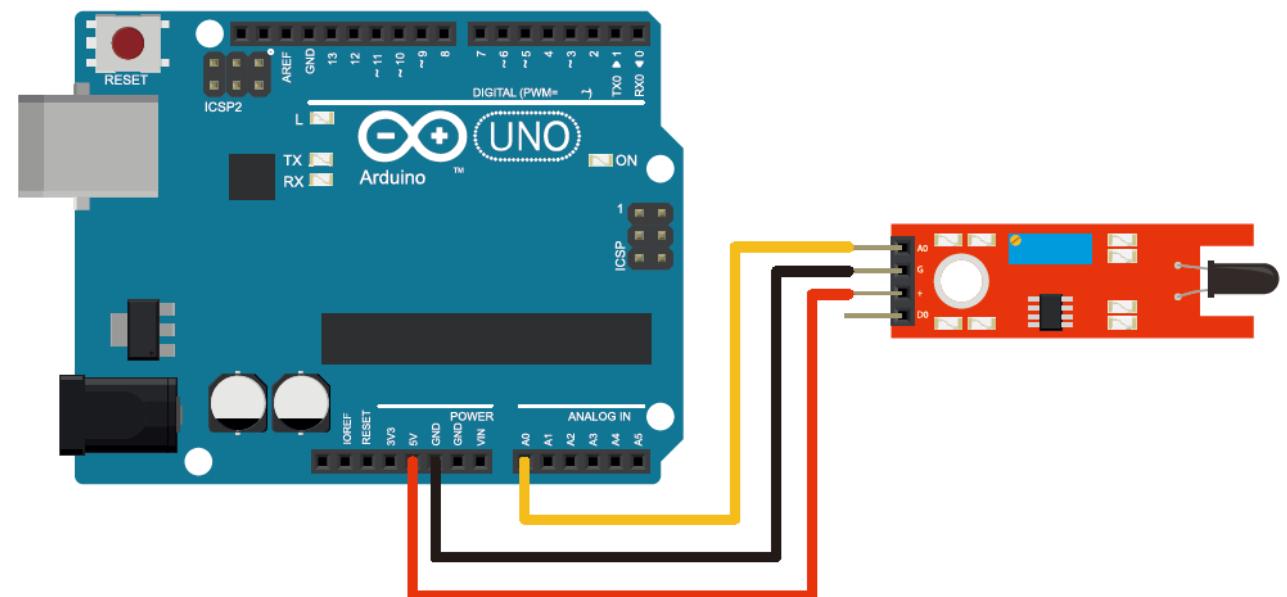
- 불꽃 감지 아두이노 실험 코드 작성

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

void loop()
{
    int analog_value = analogRead(A0);

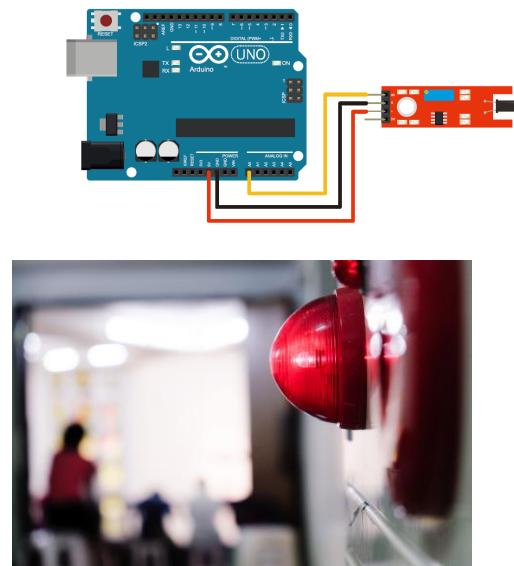
    Serial.println(analog_value);

    delay(100);
}
```



# 불꽃감지센서를 이용한 화재감지 응용

- 불꽃이 감지 되면 자동으로 경고를 발생시키자!

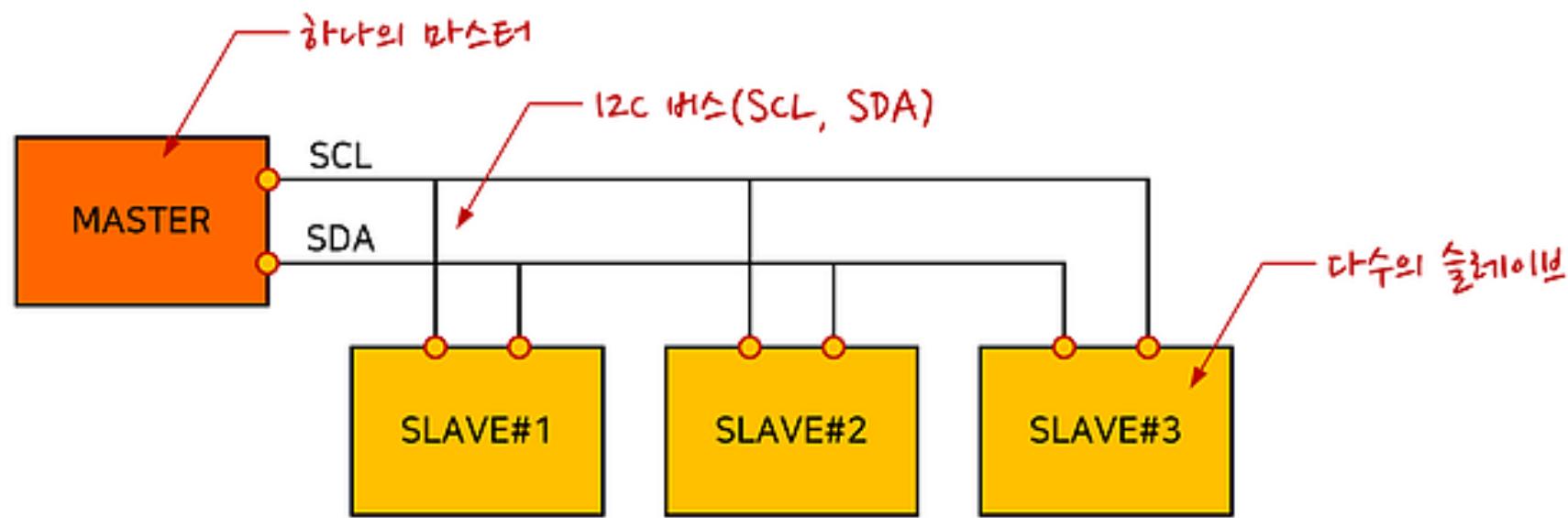


# MLX90614

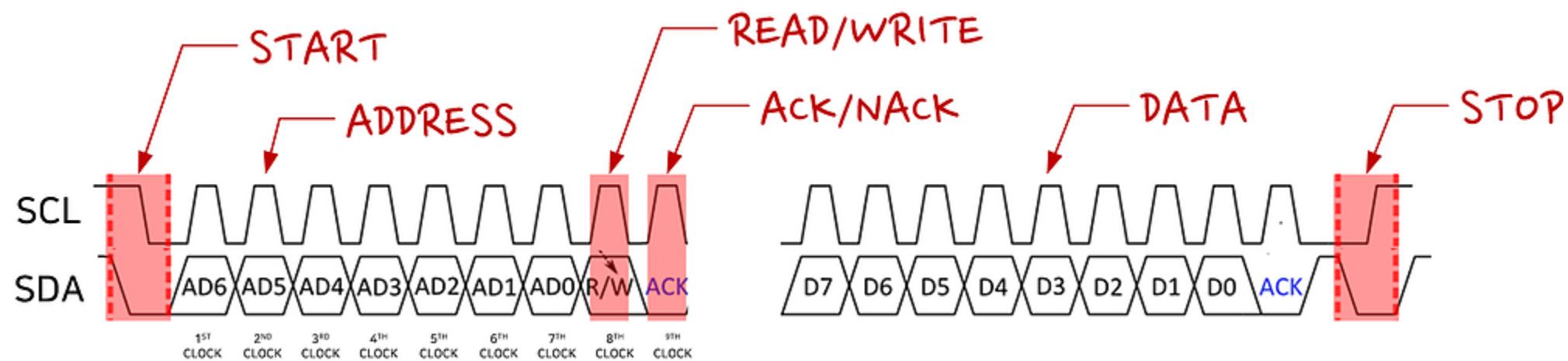
- 비접촉식 온도센서 모듈
- FOV 90°
- 측정범위 : -70°C ~ 380°C
- 인터페이스 : I2C



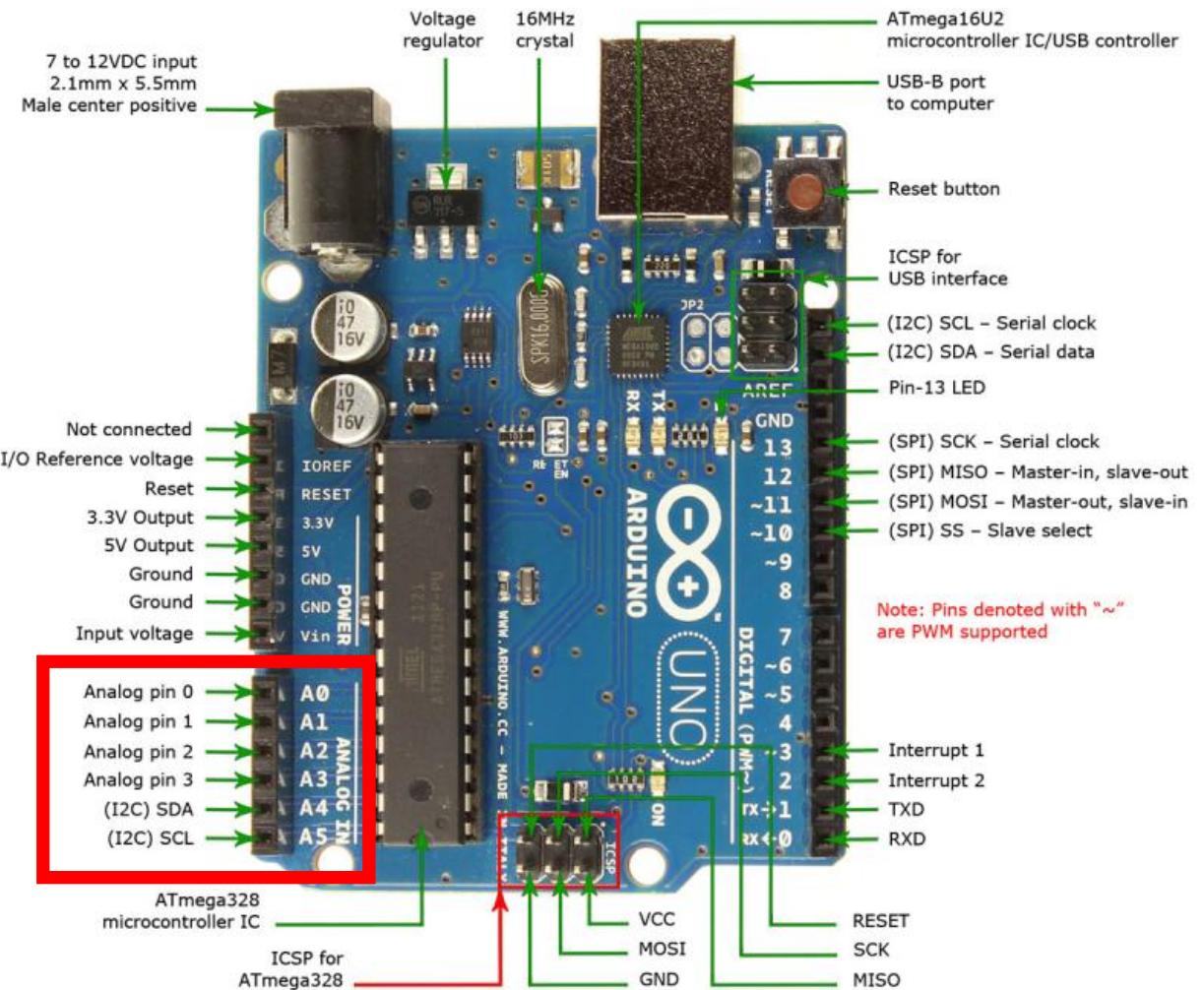
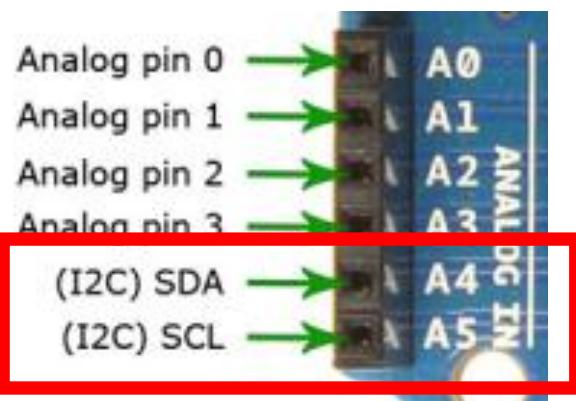
# I2C 통신



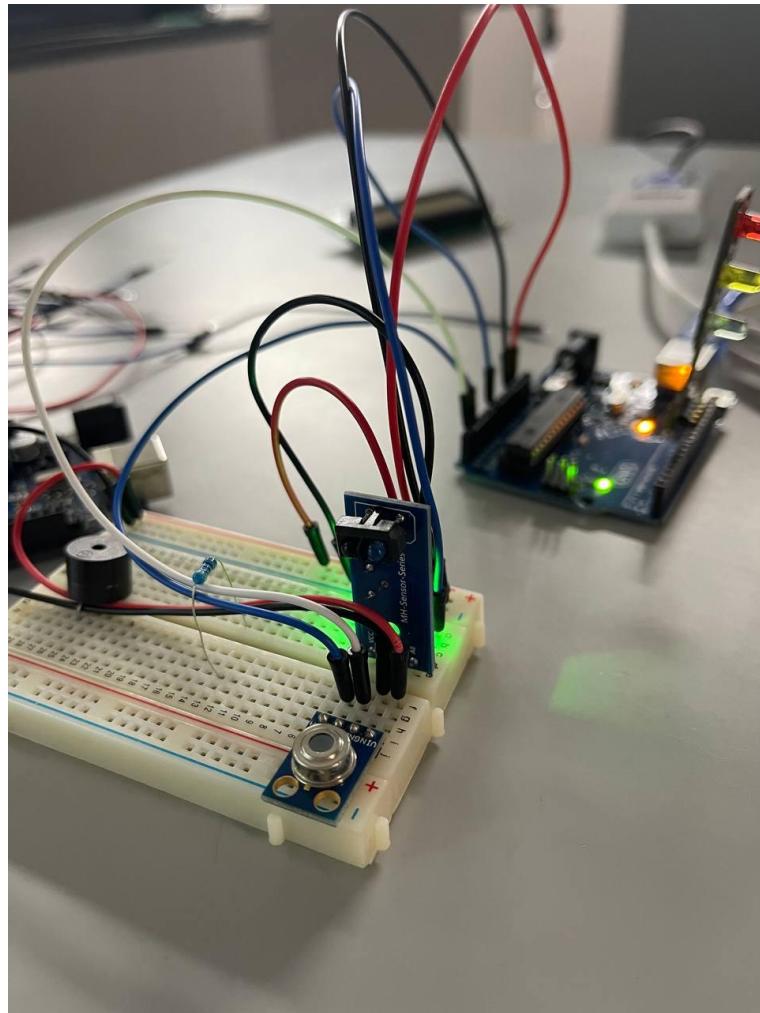
# I2C 통신



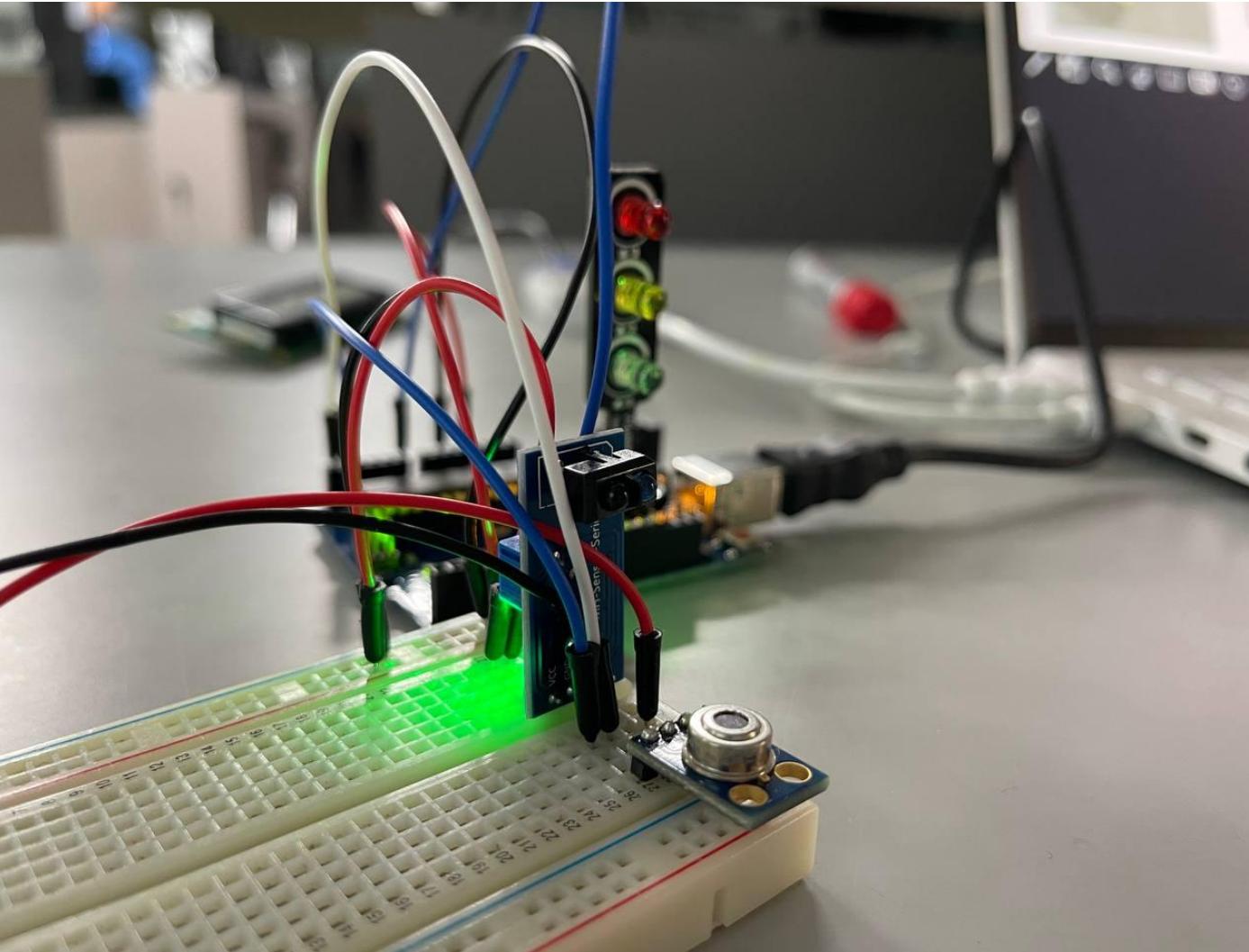
# 아두이노의 I2C통신



# MLX90614 테스트

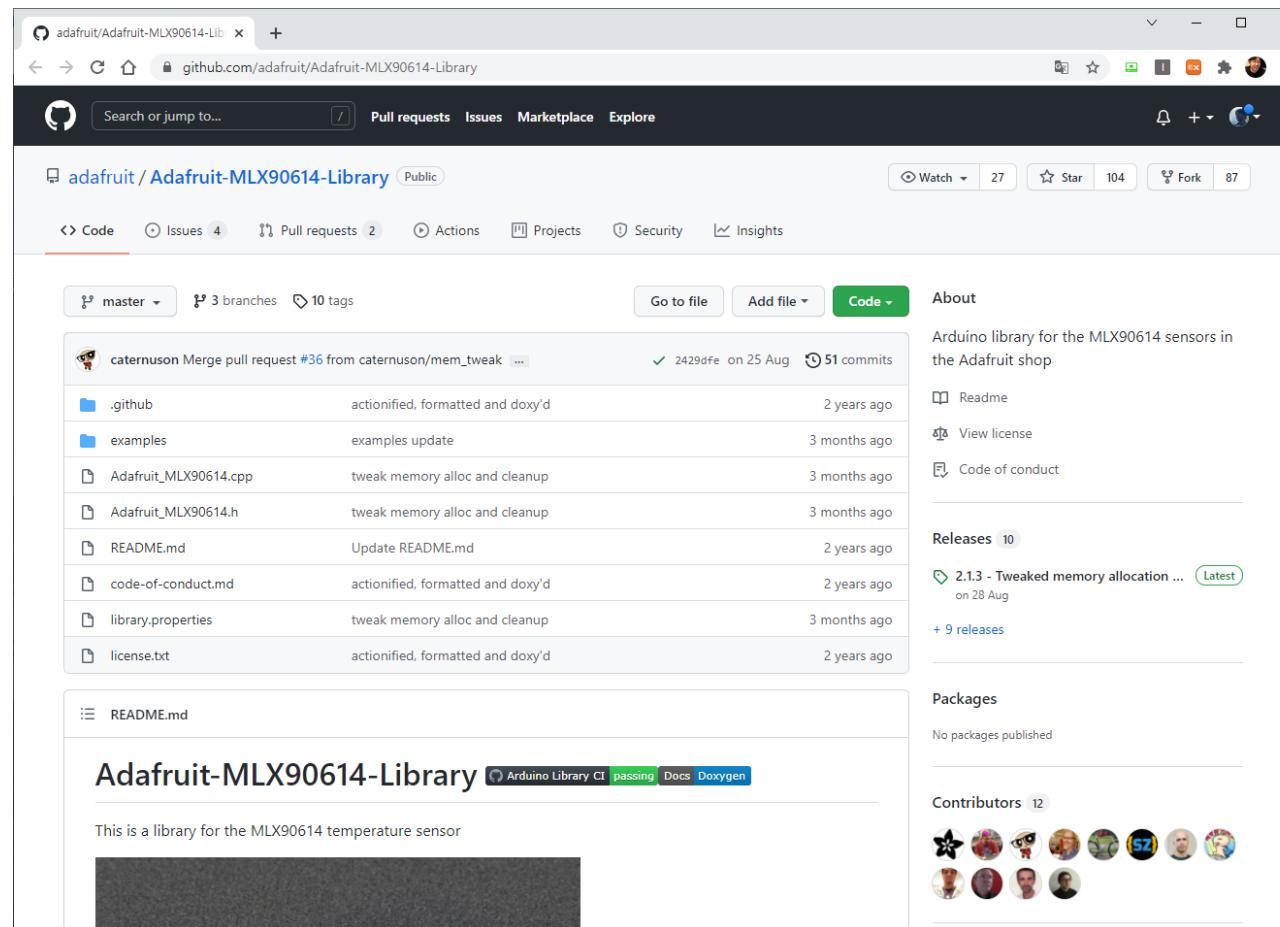


# MLX90614 테스트



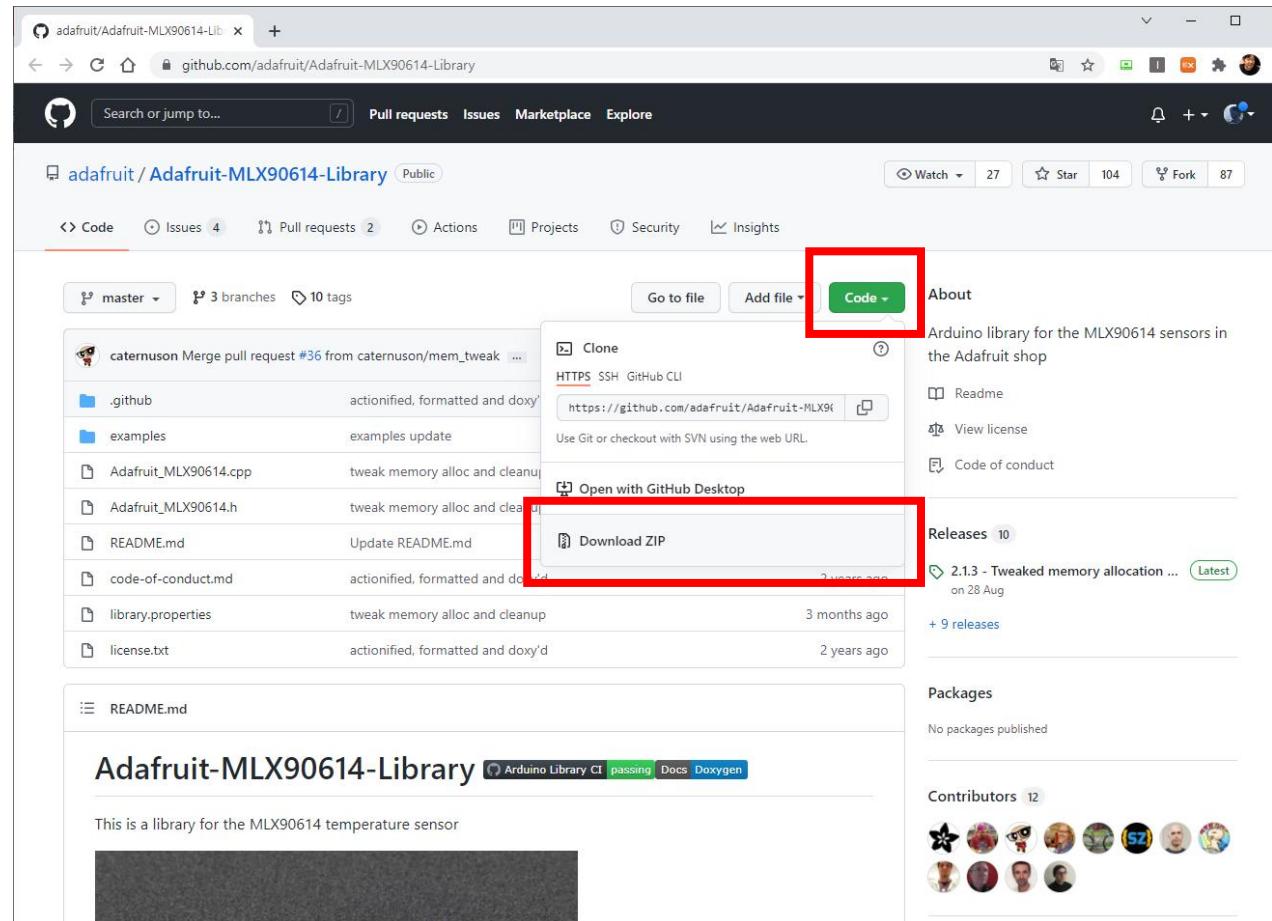
# MLX90614 라이브러리 사용

- <https://github.com/adafruit/Adafruit-MLX90614-Library>

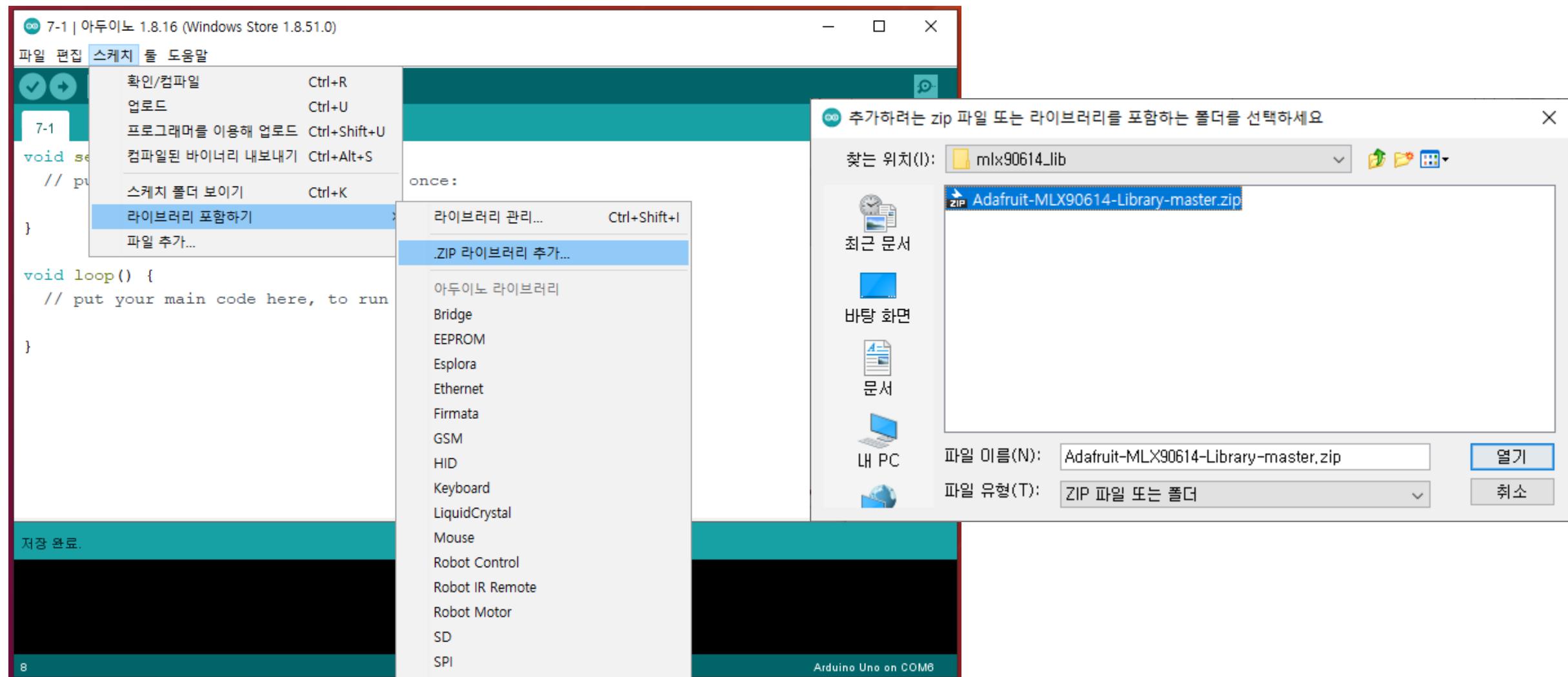


# MLX90614 라이브러리 사용

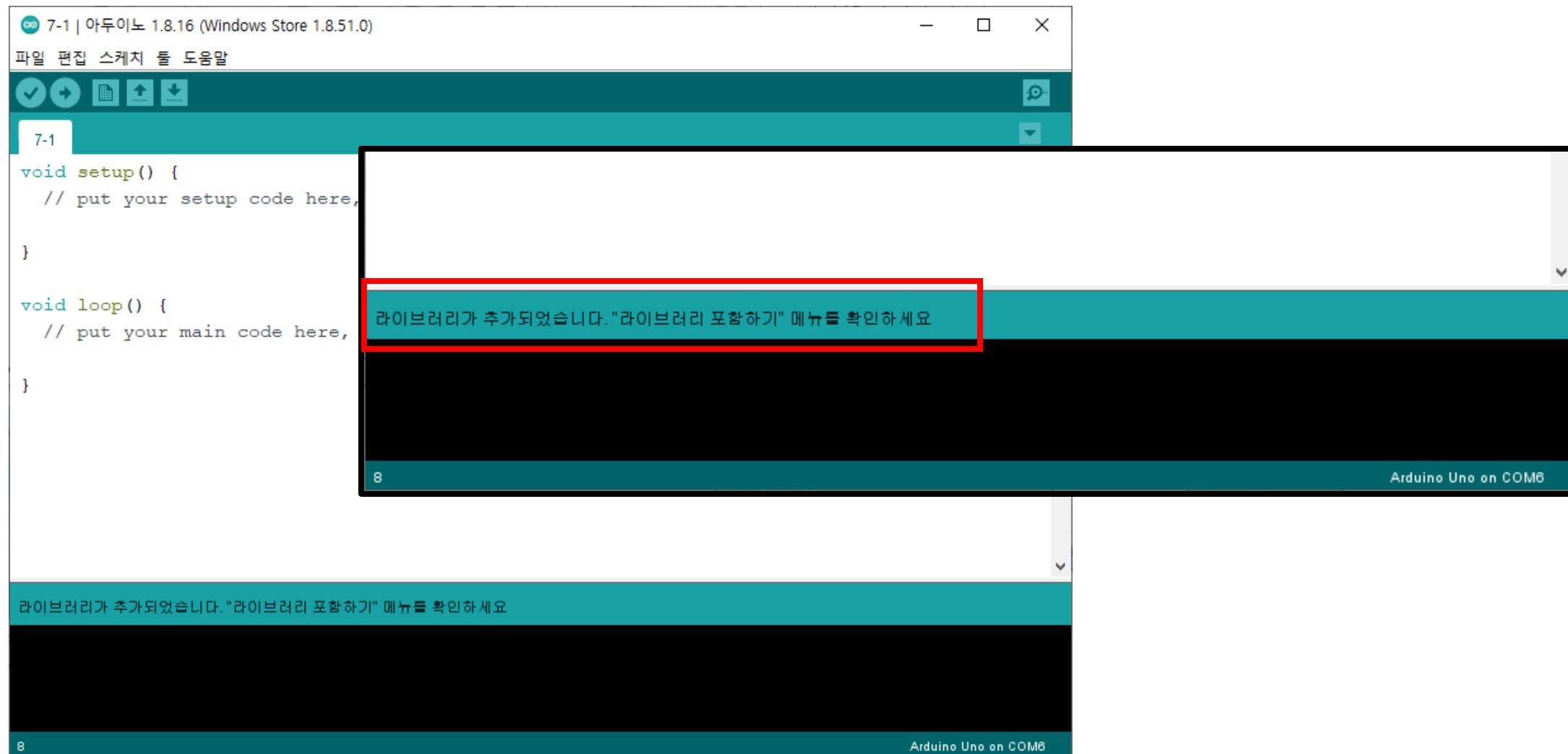
- <https://github.com/adafruit/Adafruit-MLX90614-Library>



# MLX90614 라이브러리 사용



# MLX90614 라이브러리 사용



# MLX90614 라이브러리 사용

The image shows two side-by-side screenshots of the Arduino IDE version 1.8.16 running on Windows Store 1.8.51.0.

**Left Screenshot:** A code editor window titled "7-1 | 아두이노 1.8.16 (Windows Store 1.8.51.0)". The code is as follows:

```
7-1 | 아두이노 1.8.16 (Windows Store 1.8.51.0)
파일 편집 스케치 툴 도움말
7-1
void setup() {
    // put your setup code here, to run
}
void loop() {
    // put your main code here, to run
}

라이브러리가 추가되었습니다. "라이브러리 포함하기" 메뉴를 통해 추가되었습니다.
```

A context menu is open over the line "라이브러리 포함하기". The "라이브러리 관리..." option is highlighted. Other options include ".ZIP 라이브러리 추가...", "아두이노 라이브러리", "Bridge", "EEPROM", "Esplora", "Ethernet", "Firmata", "GSM", "HID", "Keyboard", "LiquidCrystal", "Mouse", "Robot Control", "Robot IR Remote", "Robot Motor", "SD", "SPI", "Servo", "SoftwareSerial", "SpacebrewYun", "Stepper", "TFT", "Temboo", "WiFi", and "Wire". At the bottom of the menu, there are links to "추천 라이브러리", "Adafruit Circuit Playground", and "Adafruit MLX90614 Library".

**Right Screenshot:** A code editor window titled "7-1 | 아두이노 1.8.16 (Windows Store 1.8.51.0)". The code is as follows:

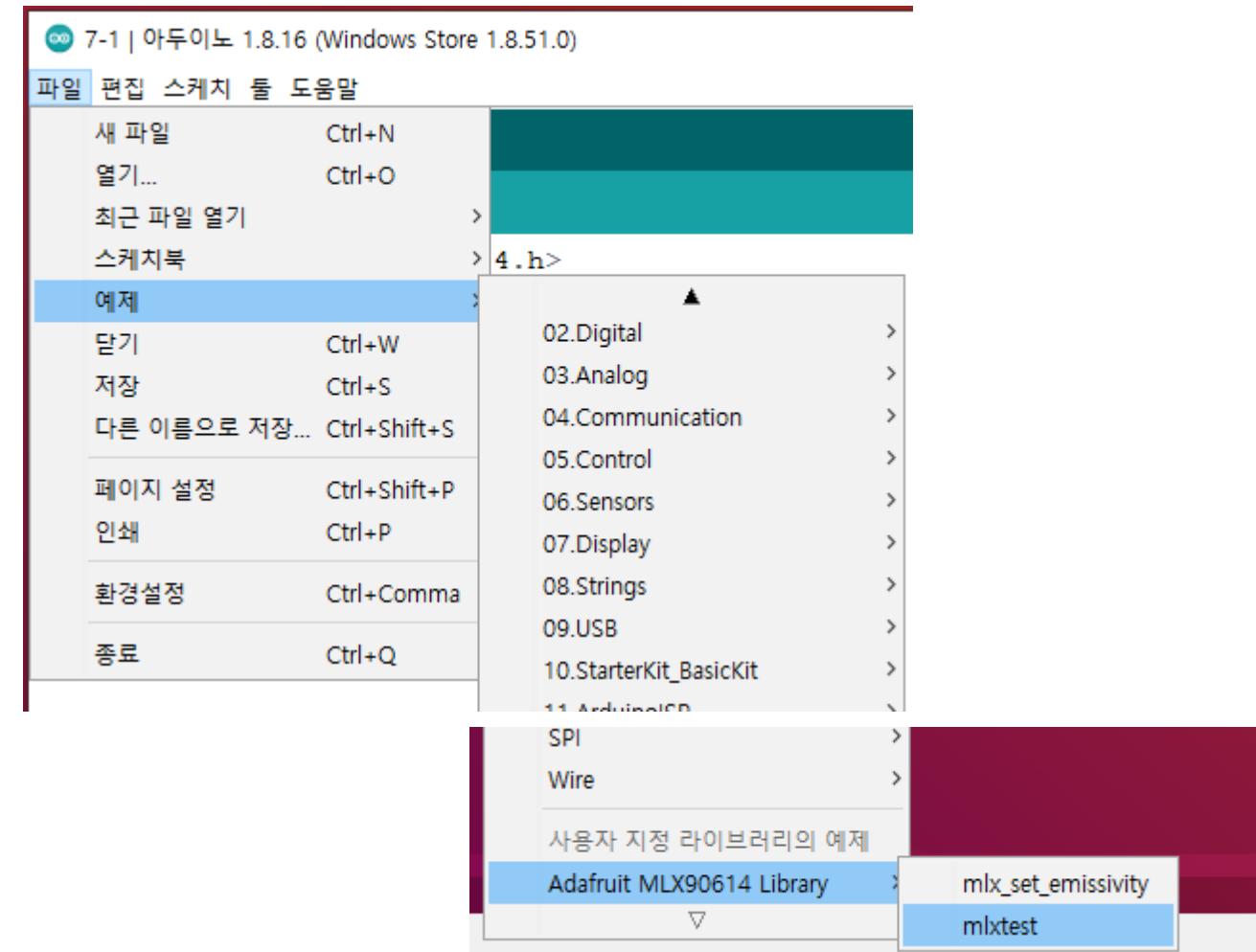
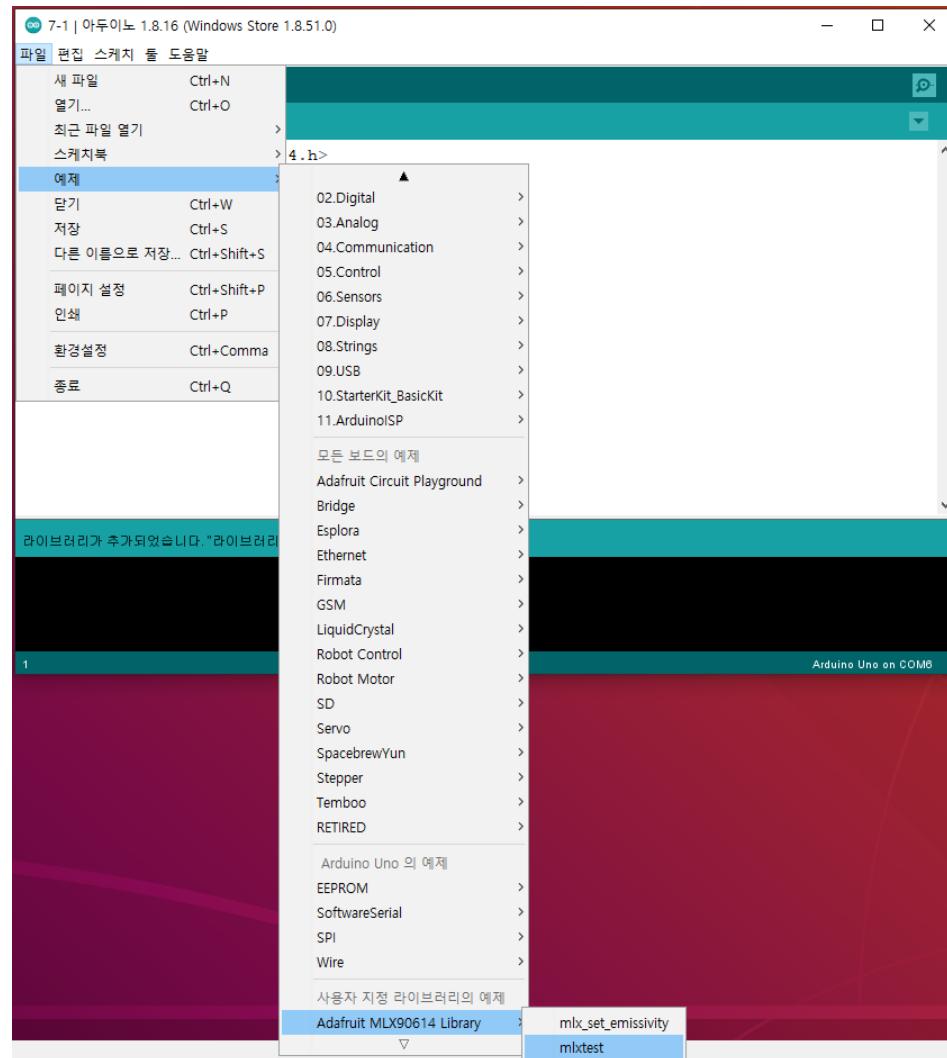
```
#include <Adafruit_MLX90614.h>

void setup() {
    // put your setup code here, to run once:
}

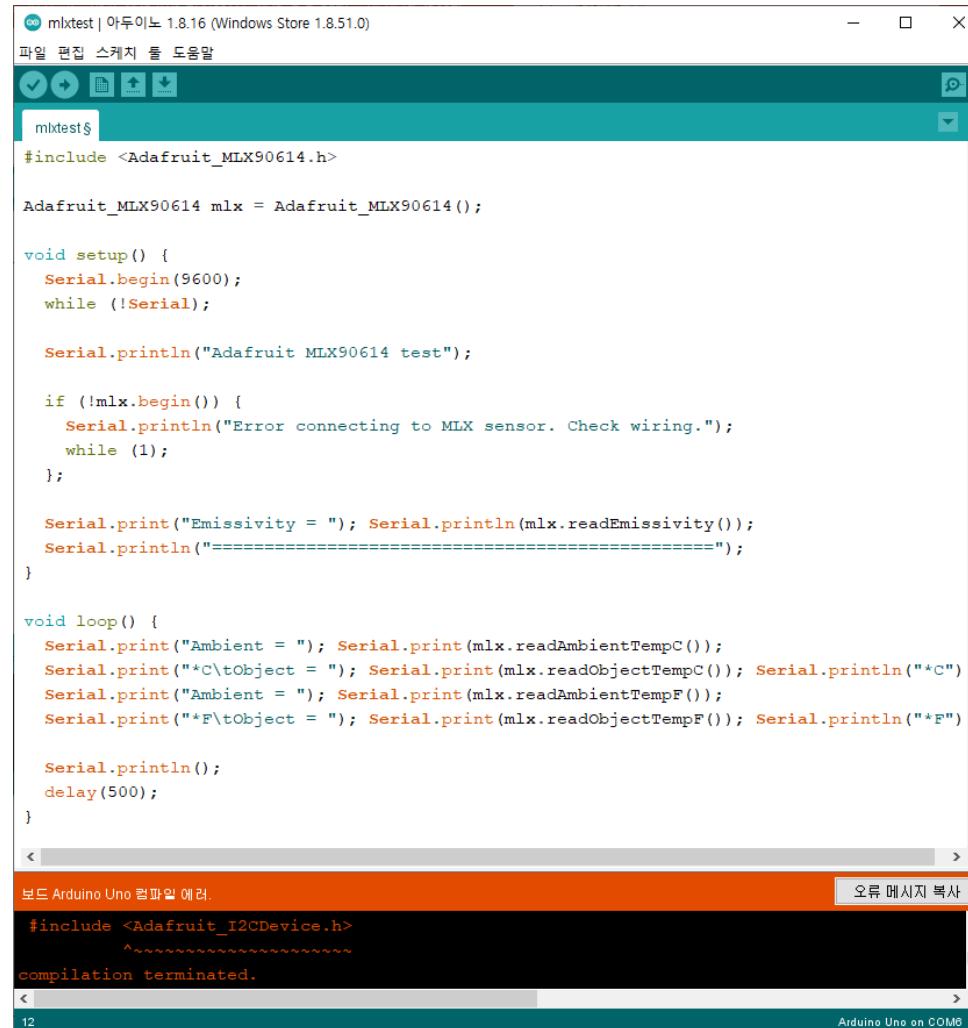
void loop() {
    // put your main code here, to run repeatedly:
}
```

The line "#include <Adafruit\_MLX90614.h>" is highlighted with a red box. Below the code, there are sections for "WiFi" and "Wire". At the bottom, there is a link to "추천 라이브러리", "Adafruit Circuit Playground", and "Adafruit MLX90614 Library".

# MLX90614 라이브러리 예제 테스트



# MLX90614 라이브러리 예제 테스트



The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** mlxtest | 아두이노 1.8.16 (Windows Store 1.8.51.0)
- File Menu:** 파일 편집 스케치 둘 도움말
- Toolbar:** Includes icons for Open, Save, Print, and others.
- Code Editor:** The code for the mlxtest sketch is displayed:

```
#include <Adafruit_MLX90614.h>

Adafruit_MLX90614 mlx = Adafruit_MLX90614();

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

  Serial.println("Adafruit MLX90614 test");

  if (!mlx.begin()) {
    Serial.println("Error connecting to MLX sensor. Check wiring.");
    while (1);
  }

  Serial.print("Emissivity = "); Serial.println(mlx.readEmissivity());
  Serial.println("=====");
}

void loop() {
  Serial.print("Ambient = "); Serial.print(mlx.readAmbientTempC());
  Serial.print("\tObject = "); Serial.print(mlx.readObjectTempC()); Serial.println("C")
  Serial.print("Ambient = "); Serial.print(mlx.readAmbientTempF());
  Serial.print("\tObject = "); Serial.print(mlx.readObjectTempF()); Serial.println("F")

  Serial.println();
  delay(500);
}
```
- Serial Monitor:** Shows the message "보드 Arduino Uno 컴파일 에러." (Board Arduino Uno compilation error.)
- Bottom Status Bar:** Shows "Arduino Uno on COM6".



The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** 보드 Arduino Uno 컴파일 에러.
- Code Editor:** The code for the mlxtest sketch is displayed:

```
#include <Adafruit_I2CDevice.h>
^~~~~~
```
- Message Area:** Shows the error message "compilation terminated."

# MLX90614 라이브러리 예제 테스트

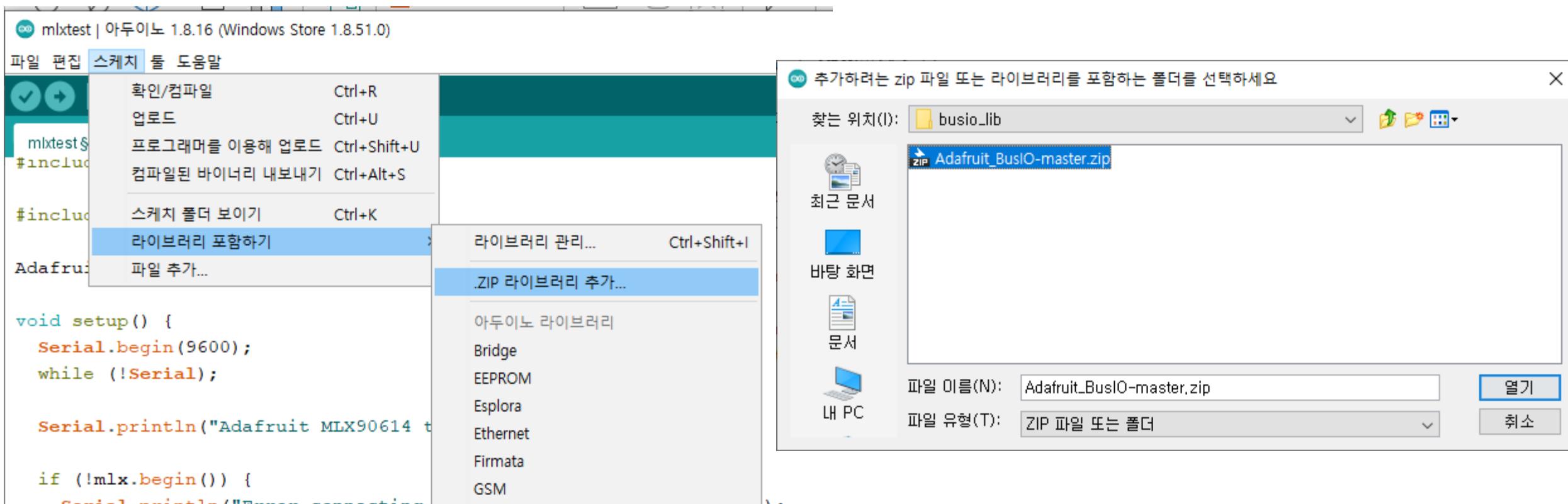
- [https://github.com/adafruit/Adafruit\\_BusIO](https://github.com/adafruit/Adafruit_BusIO) 라이브러리 다운로드

This screenshot shows the GitHub repository page for Adafruit\_BusIO. The main navigation bar at the top includes links for Pull requests, Issues, Marketplace, and Explore. Below the header, there's a search bar and a navigation bar with tabs for Code, Issues (2), Pull requests, Actions, Projects, Wiki, Security, and Insights. The Code tab is currently selected. On the left, a sidebar lists branches (master, 7 branches), tags (48 tags), and recent commits. The main content area displays the repository's README.md file, which contains a brief description of I2C, SPI, and UART abstractions for Arduino, a Readme link, and an MIT License link. It also features sections for Releases (48 releases, with the latest being 1.9.3), Packages, and Contributors (21 contributors). At the bottom, there's a footer with the text "Adafruit Bus IO Library" and "Arduino Library CI passing".

This screenshot shows the same GitHub repository page as the first one, but with a red box highlighting the 'Code' button in the top right corner of the main content area. A dropdown menu has been opened from this button, showing options: 'Clone' (with HTTPS, SSH, and GitHub CLI links), 'Open with GitHub Desktop', and 'Download ZIP'. The rest of the page content is identical to the first screenshot.

# MLX90614 라이브러리 예제 테스트

- [https://github.com/adafruit/Adafruit\\_BusIO](https://github.com/adafruit/Adafruit_BusIO) 라이브러리 추가



# MLX90614 라이브러리 예제 테스트

