

101111000101001010011000101001111010101010111110011111010100001111100001000010101011001111101010000111110000100001010101111100111110101000011111000010000101011001111101010000111110000100001010110011111010100001111100001000010101010000101

LCD 란

■ Liquid Crystal Display

■ 액정의 전기적 성질을 이용하여 시각적인 효과를 주는 전자 제품



예시1. 16x2 C-LCD 모듈



예시2. 계산기



예시3. 스톱워치

C-LCD 사용해보기

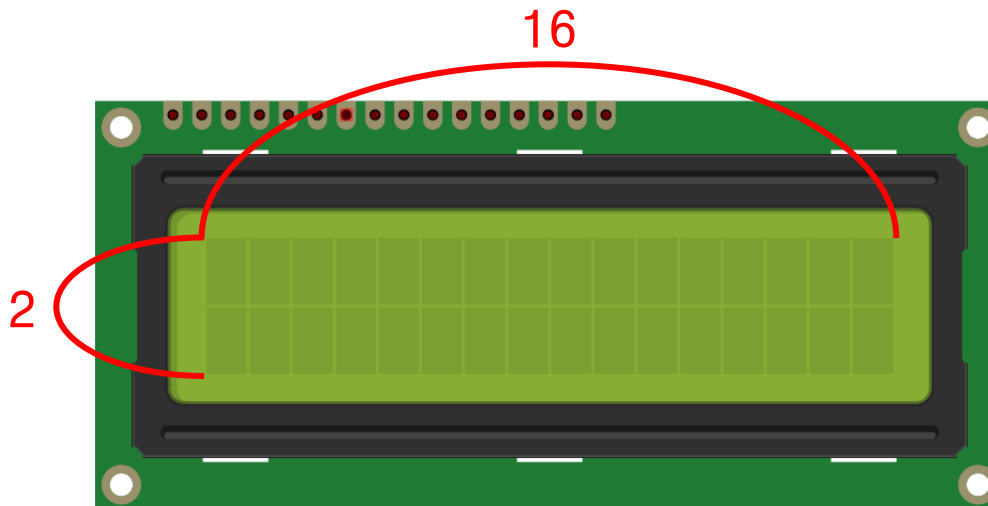
■ C-LCD: Character LCD

■ 한 화면에 가로 16, 세로 2의 총 32개의 문자 표시가능

■ I2C 통신 방법을 사용하며 주소를 확인이 필요할 수도 있다

❖ 고유 주소 확인 필요

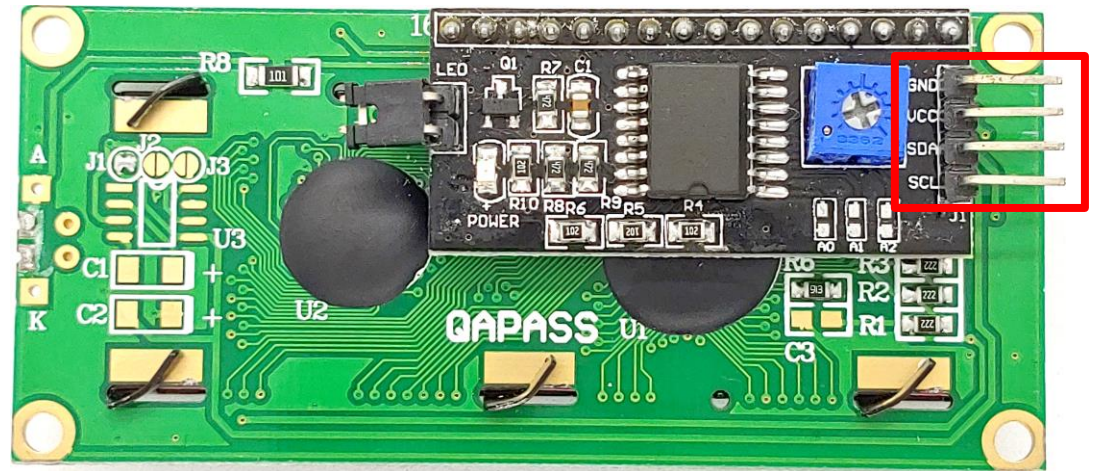
❖ I2C는 GPIO를 SDA/SCL 로 사용한다



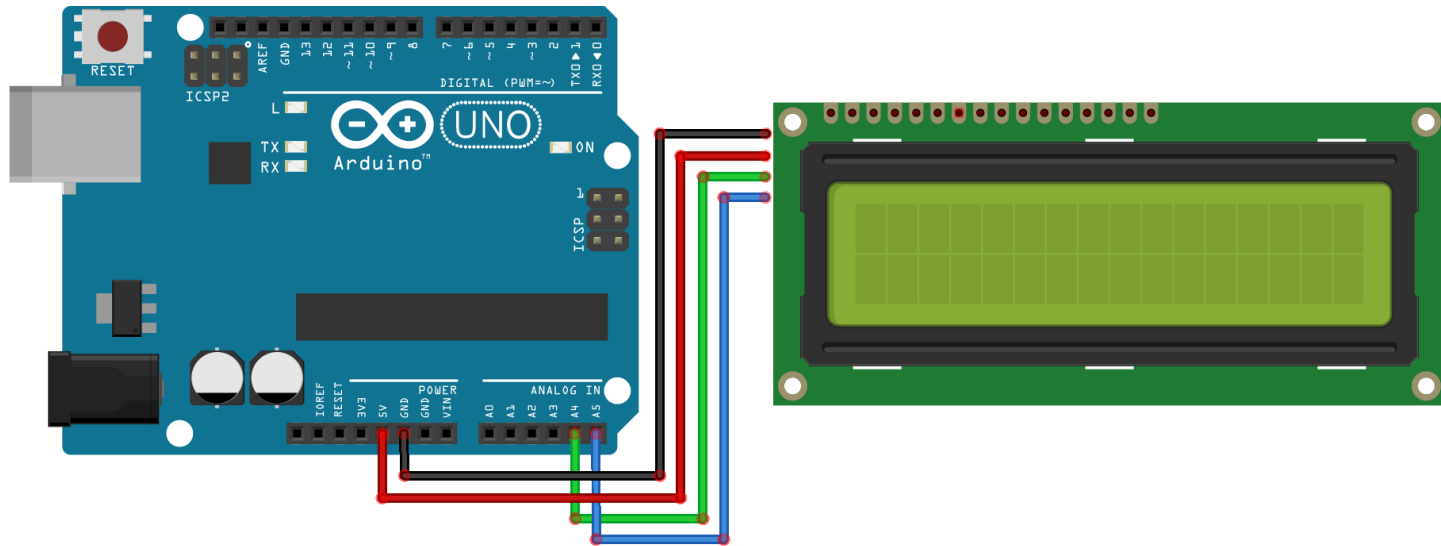
C-LCD 모듈 핀 아웃

■ 다음과 같이 연결

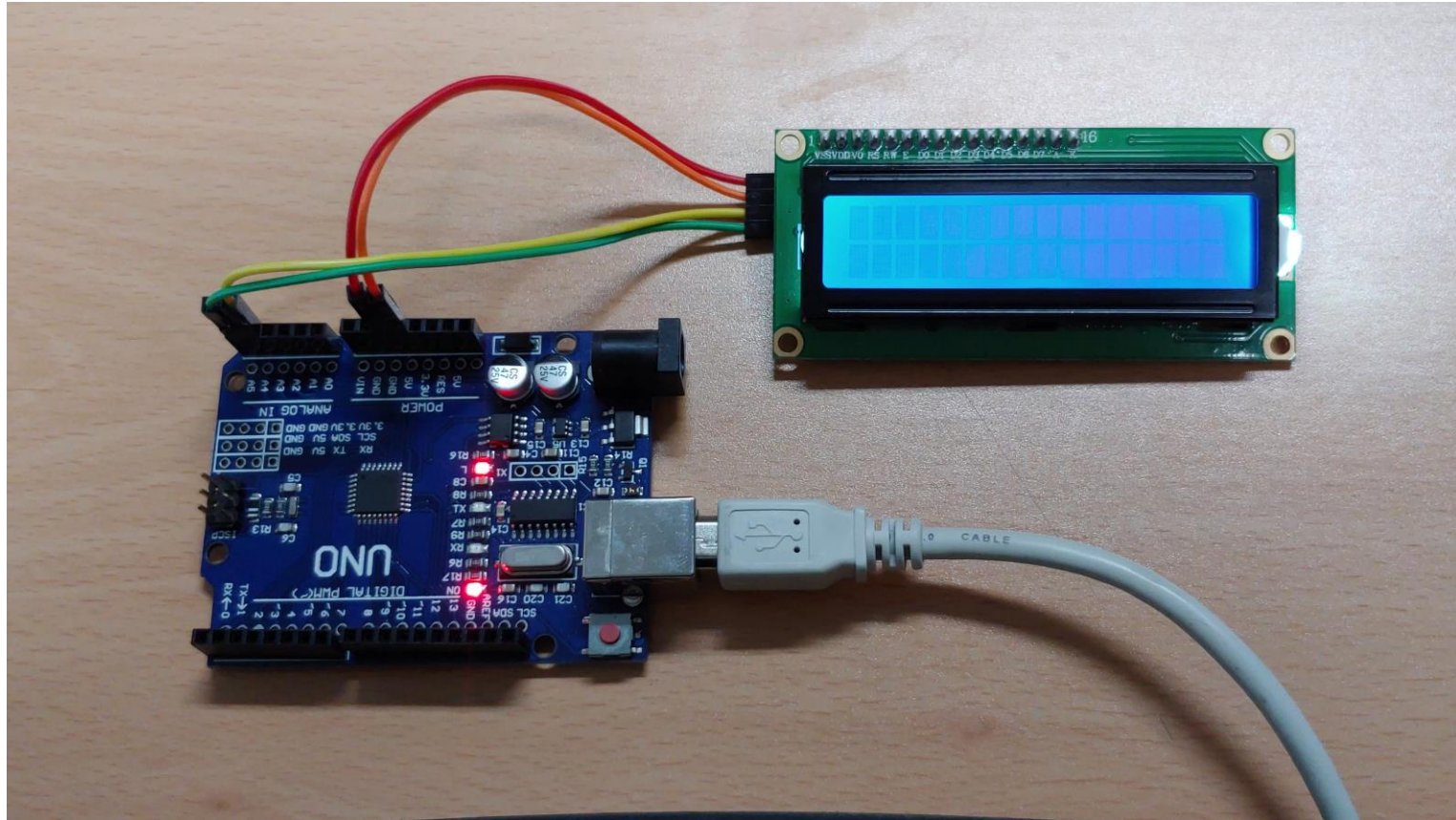
- ❖ GND – GND
- ❖ VCC – 5V
- ❖ SDA – A4
- ❖ SCL – A5



연결도



동작 확인



예제 코드 – LCD

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

LiquidCrystal_I2C lcd(0x3F,16,2);

void setup(){
  lcd.init();
  lcd.backlight();
}

void loop(){
  lcd.print("First line!");
  lcd.setCursor(0,1);
  lcd.print("Second line!");
  delay(1000);
  lcd.clear();
  delay(1000);
}
```

함수소개 – LiquidCrystal_I2C

■ LiquidCrystal_I2C lcd(address, col, row)

■ I2C 캐릭터 LCD 설정을 정의하는 함수

❖ address : I2C 디바이스 주소

❖ col : LCD의 열 개수

❖ Row : LCD의 행 개수

❖ ex) 0x3f 주소의 16x2 I2C 캐릭터 LCD 사용시
LiquidCrystal_I2C lcd(0x3f, 16, 2)

함수소개 – LiquidCrystal 함수

■ lcd.init(), lcd.backlight()

❖ LCD 초기화 하는 함수 setup 루프에 한번 선언 해주면 된다.

■ lcd.clear()

❖ LCD를 비우는 함수. LCD개의 문자를 전부 지운다.

❖ 커서의 위치도 0,0으로 초기화 됨

■ lcd.print(data);

❖ LCD에 data를 출력하는 함수.

❖ data : 출력 할 데이터

함수소개 – lcd.setCursor

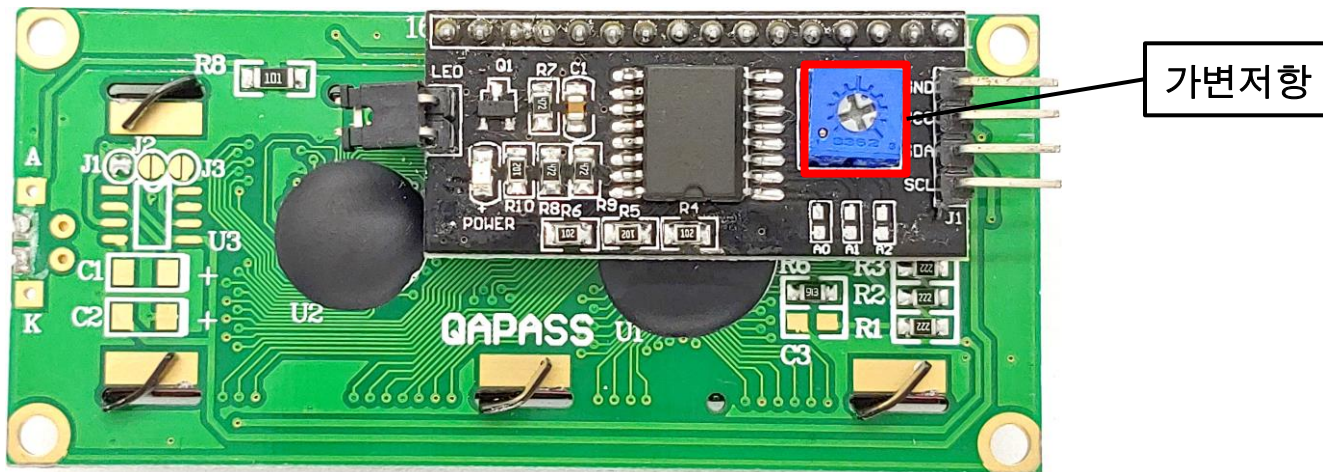
■ lcd.setCursor(col, row)

- ❖ 해당 위치로 커서를 옮길 때 쓰는 함수.
- ❖ lcd.print 함수 사용시 0,0을 기준으로 하기 때문에 다른 줄에 내용을 입력하고자 할 때 사용
- ❖ col : 설정할 열의 위치
- ❖ row : 설정할 행의 위치

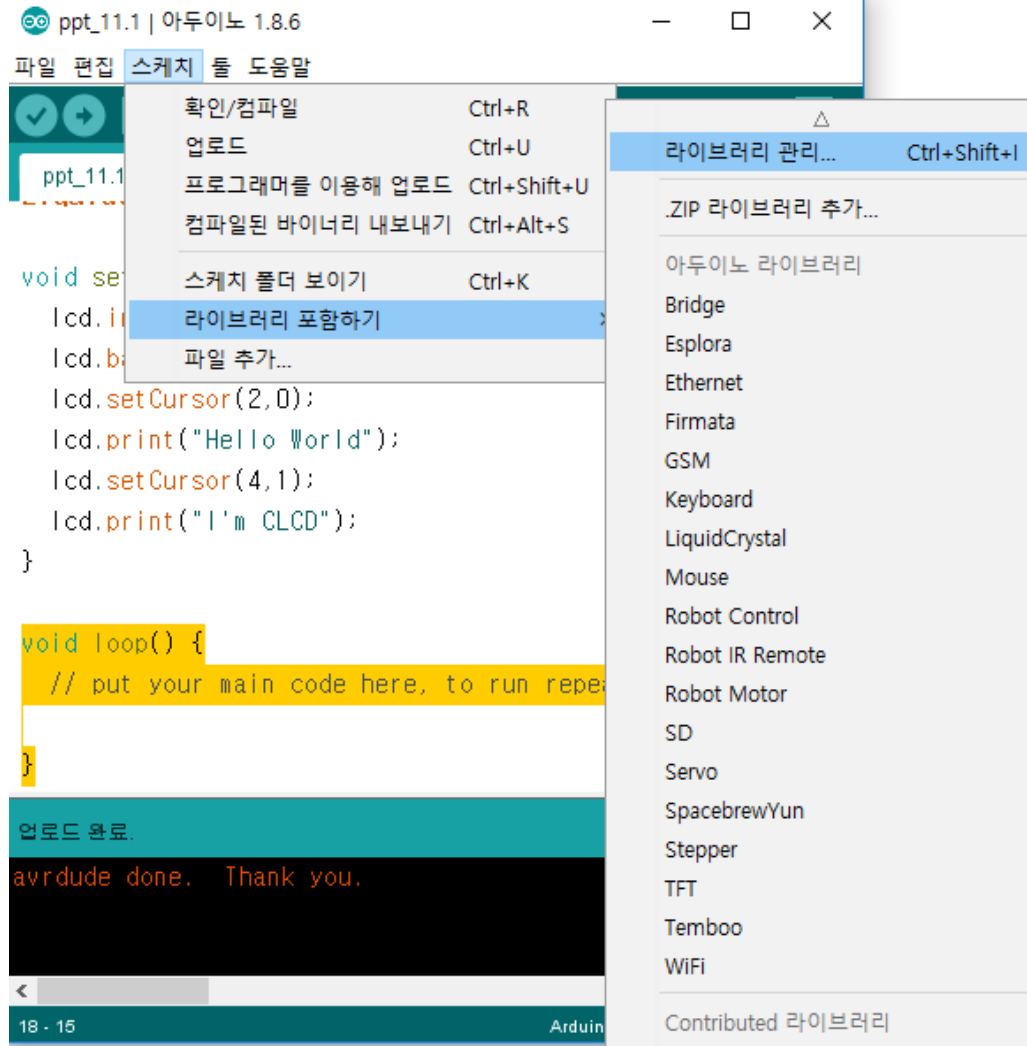
- ❖ ex) 두 번째 줄 첫 번째 열로 커서 설정 시
lcd.setCursor(0,1);

명암(선명도) 조절하기.

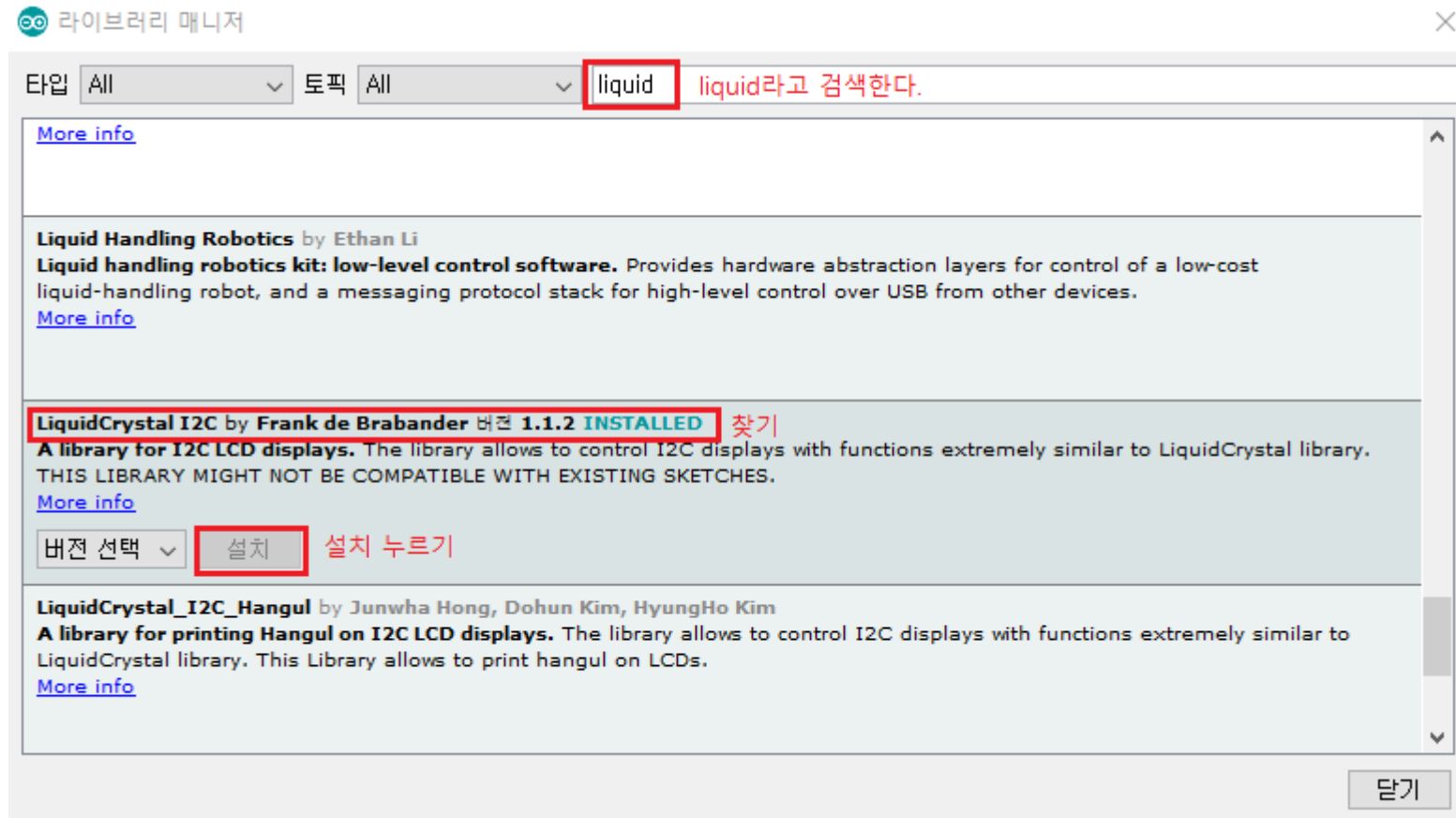
- 명암(선명도)이 너무 낮으면 텍스트가 보이지 않고, 너무 높으면 잔상이 오래 남기 때문에 명암을 적당하게 조절해야 할 필요가 있다.
- I2C모듈에 내장 되어있는 가변저항을 돌려서 텍스트의 명암을 조절할 수 있다.
 - ❖이 때 LCD 자체를 밝게 해주는 LED 백라이트와 혼동하면 안된다.



스케치 사용 시 C-LCD 라이브러리 설정



스케치 사용 시 C-LCD 라이브러리 설정



과제 – LCD 에 학번과 이름을 표시해보자

- 첫 번째 라인에 학번, 두 번째 라인에 이름을 표시해보자
 - ❖ 코드를 수정한다.
- LCD는 한글 폰트가 없으므로 이름은 영문이름으로 표시해보자
- 수정한 코드를 제출하시오.

1011110001010010100110001010011110101010101111100111110101000011111000010000101010110011111010100001111100001000010101011111001111101010000111110000100001010101111100111110101000011111000010000101011001111101010000111110000100001010101010000101

OLED 란

- Organic Light Emitting Diodes의 약자.
- 전류가 흐를 때 스스로 빛을 내는 유기 물질을 활용해 빛을 내는 방식. ‘자체발광 디스플레이’에 속한다.
- OLED는 화질(색표현력, 명암비 등), 무게, 두께, 저소비전력의 장점을 갖고 있으며, 유연하게 구부러지는 플렉시블의 특징을 갖고 있다.
- 백라이트를 통해 빛을 공급받아 표현하는 LCD와는 방식이 다르다.

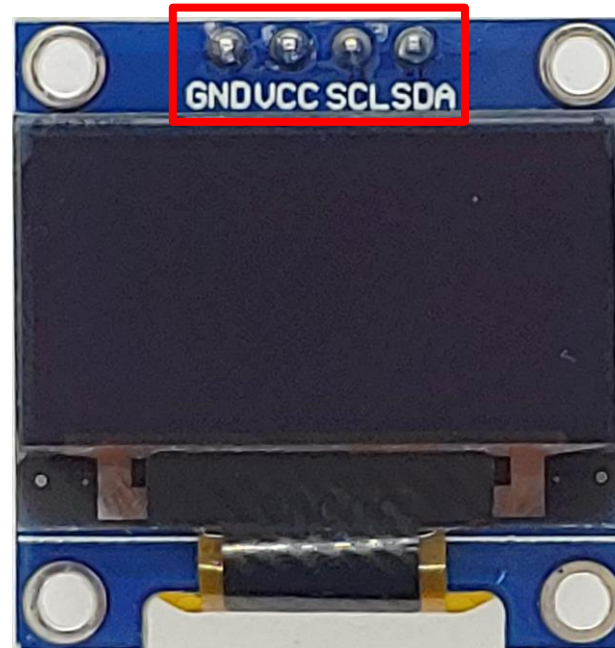
Adafruit 그래픽 라이브러리

- 일반적으로 많이 사용되는 커스텀 라이브러리.
- 2개로 구성
 - ❖ 공통된 드로잉 함수를 제공하는 GFX
 - ❖ 드라이버 칩에 따라 적절하게 동작하도록 하는 드라이버 라이브러리
- 현재 아두이노 웹 에디터 서비스에서의 커스텀 라이브러리는 아두이노 팀에서 직접 개발하거나 공동으로 개발한 라이브러리를 제외한 모든 라이브러리는 커스텀 라이브러리로 간주하여 웹 에디터에서 사용이 불가능하다. 유료플랜으로 업그레이드를 하면 커스텀 라이브러리를 사용할 수 있다.
- 유료로 업그레이드 하지 않아도 스케치 프로그램을 이용하여 커스텀 라이브러리를 사용할 수 있다.

OLED 모듈 핀 아웃

■ 다음과 같이 연결

- ❖ GND – GND
- ❖ VCC – 5V
- ❖ SCL – A5
- ❖ SDA – A4

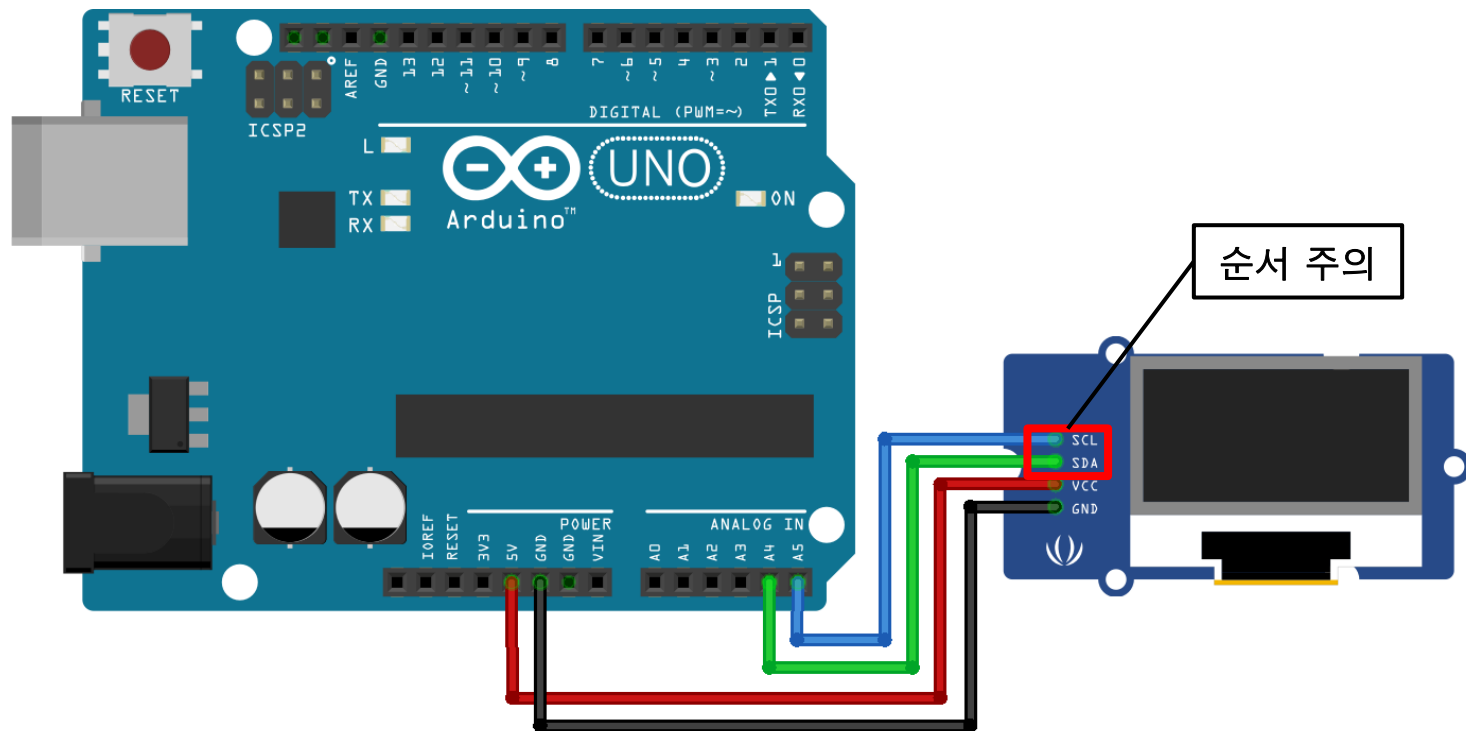


사용 예시

- OLED에 'INHATC' 를 출력해보자.
- 글자는 폰트를 사용해서 출력하게 된다.
- OLED 또 한 I2C통신 방법을 사용하며, 라이브러리 추가가 필요하다.

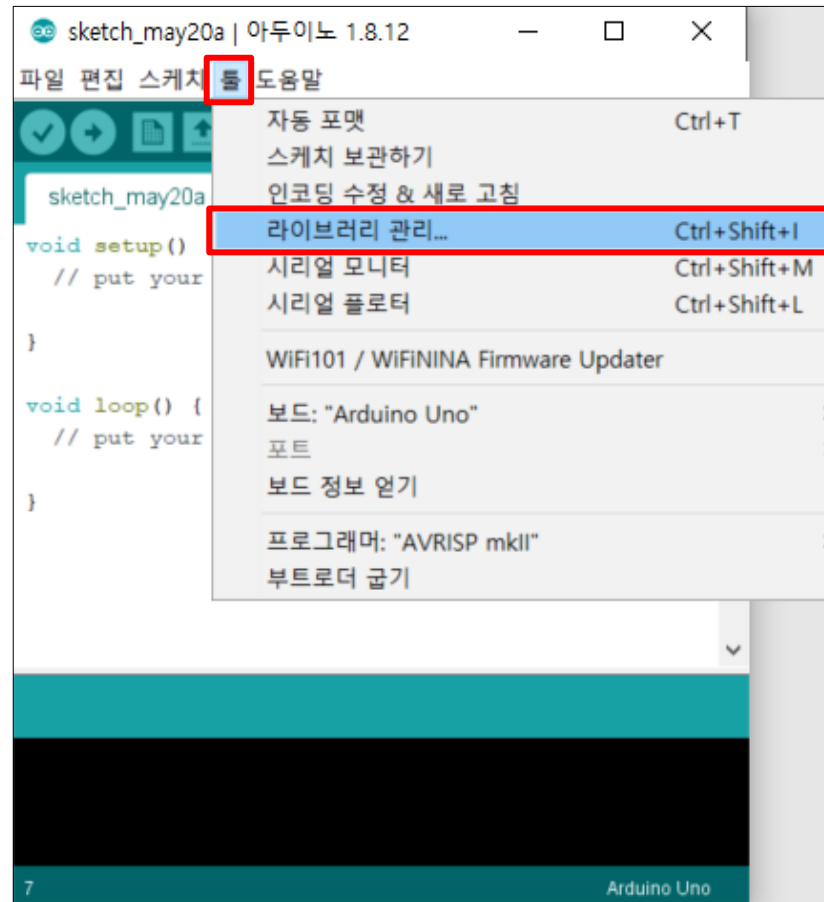


연결도



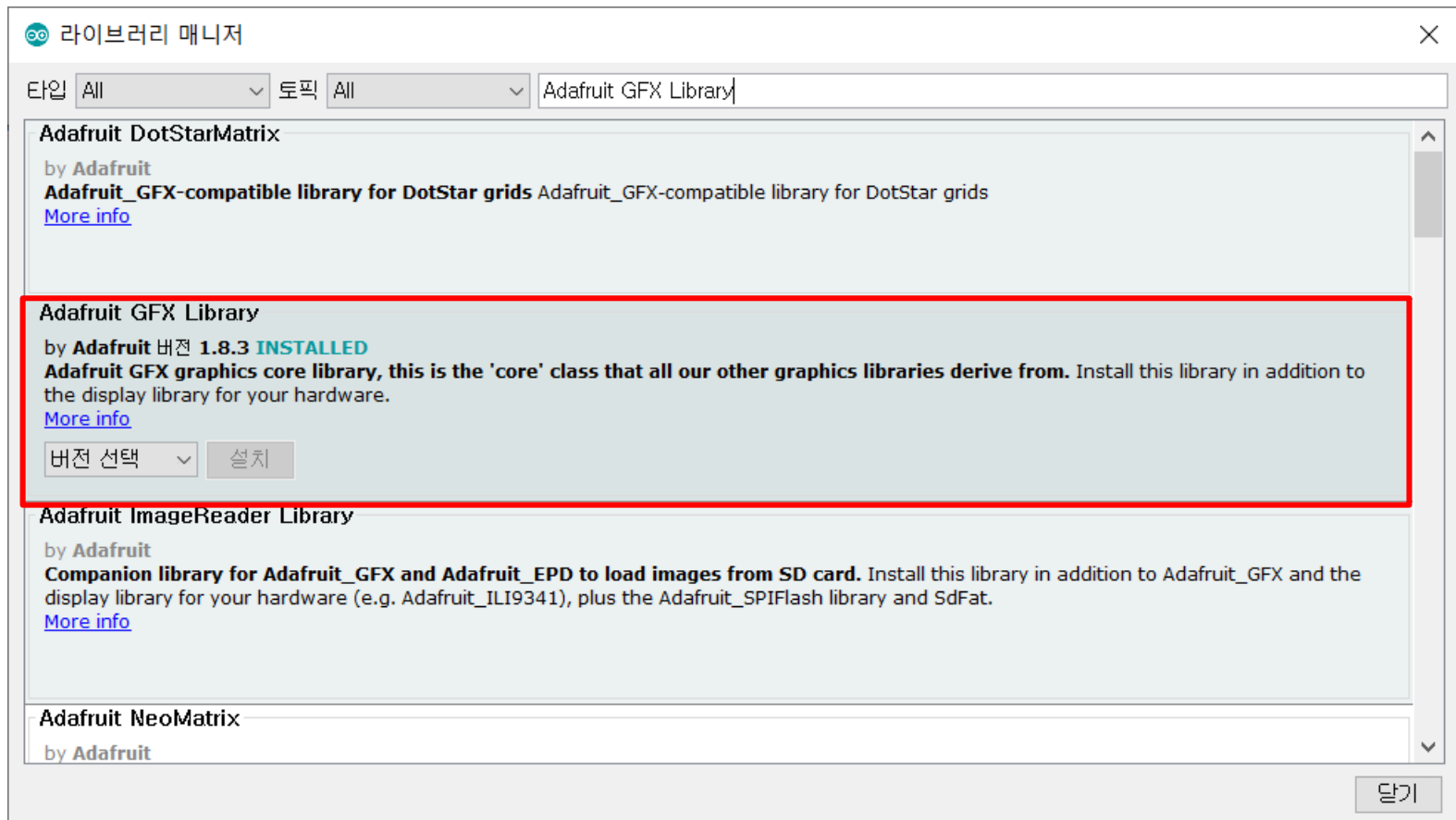
스케치에서 OLED 라이브러리 추가

■ [툴] - [라이브러리 관리...] 클릭



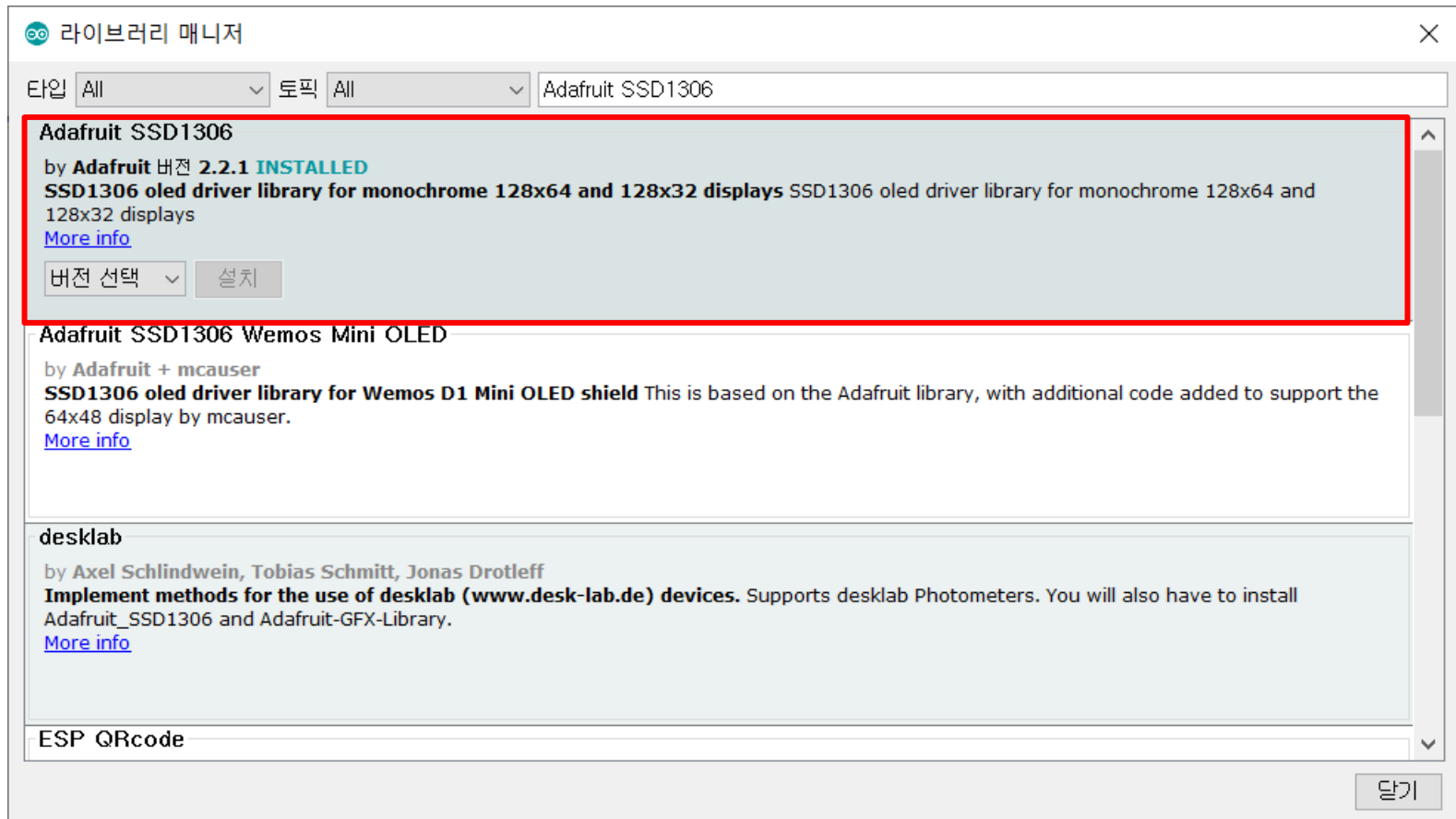
스케치에서 OLED 라이브러리 추가

■ “Adafruit GFX Library” 검색 후 설치



스케치에서 OLED 라이브러리 추가

■ “Adafruit SSD1306” 검색 후 설치



예제 코드 – 문자 출력

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

#define OLED_RESET 4

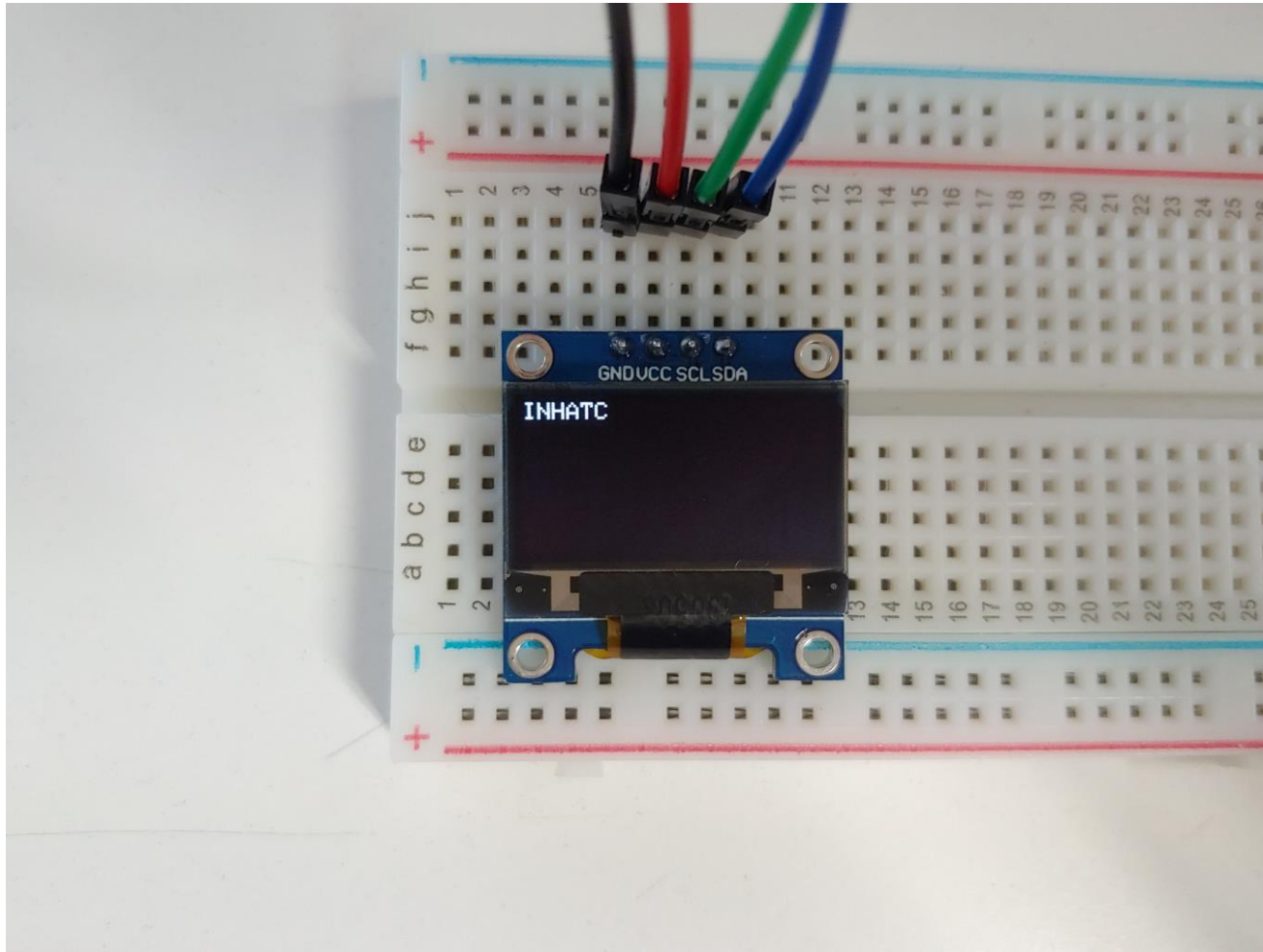
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

void setup() {
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
}

void loop() {
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0,0);
  display.println("INHATC");
  display.display();
}
```

I2C 주소확인 후 입력

동작 확인 - 문자 출력



함수소개

■ `display.begin();`

❖ OLED 초기화 하는 함수 `setup`에 한번 선언 해주면 된다.

■ `display.clearDisplay();`

❖ OLED를 비우는 함수.

❖ 커서의 위치도 0,0으로 초기화 됨

■ `display.display();`

❖ 디스플레이에 표시

함수소개

■ `display.setTextSize();`

❖ 텍스트 크기 설정

■ `display.setTextColor();`

❖ 텍스트 색 설정

■ `display.setCursor();`

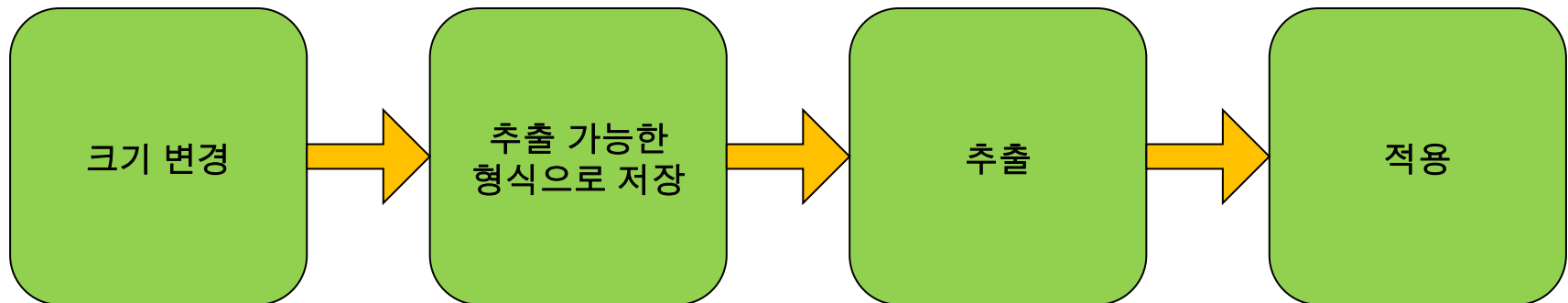
❖ 텍스트 위치 설정

■ `display.println();`

❖ 텍스트 출력 후 줄바꿈

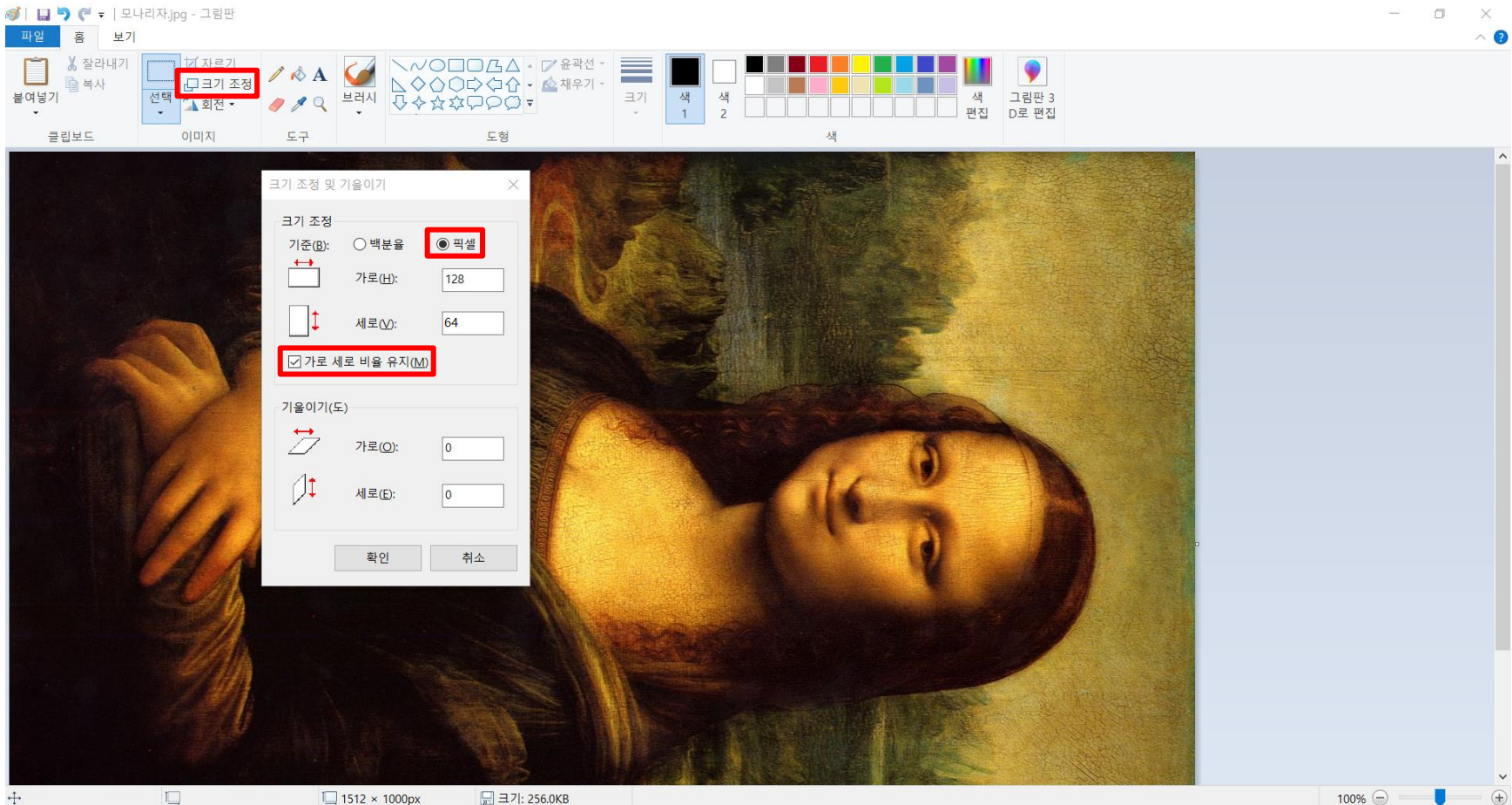
예제 - 그림 출력하기

- OLED 화면에 그림을 출력해보자.
- 그림파일을 비트맵 코드로 변경하여 출력해야 한다.
 - ❖ 작업 순서는 아래와 같다



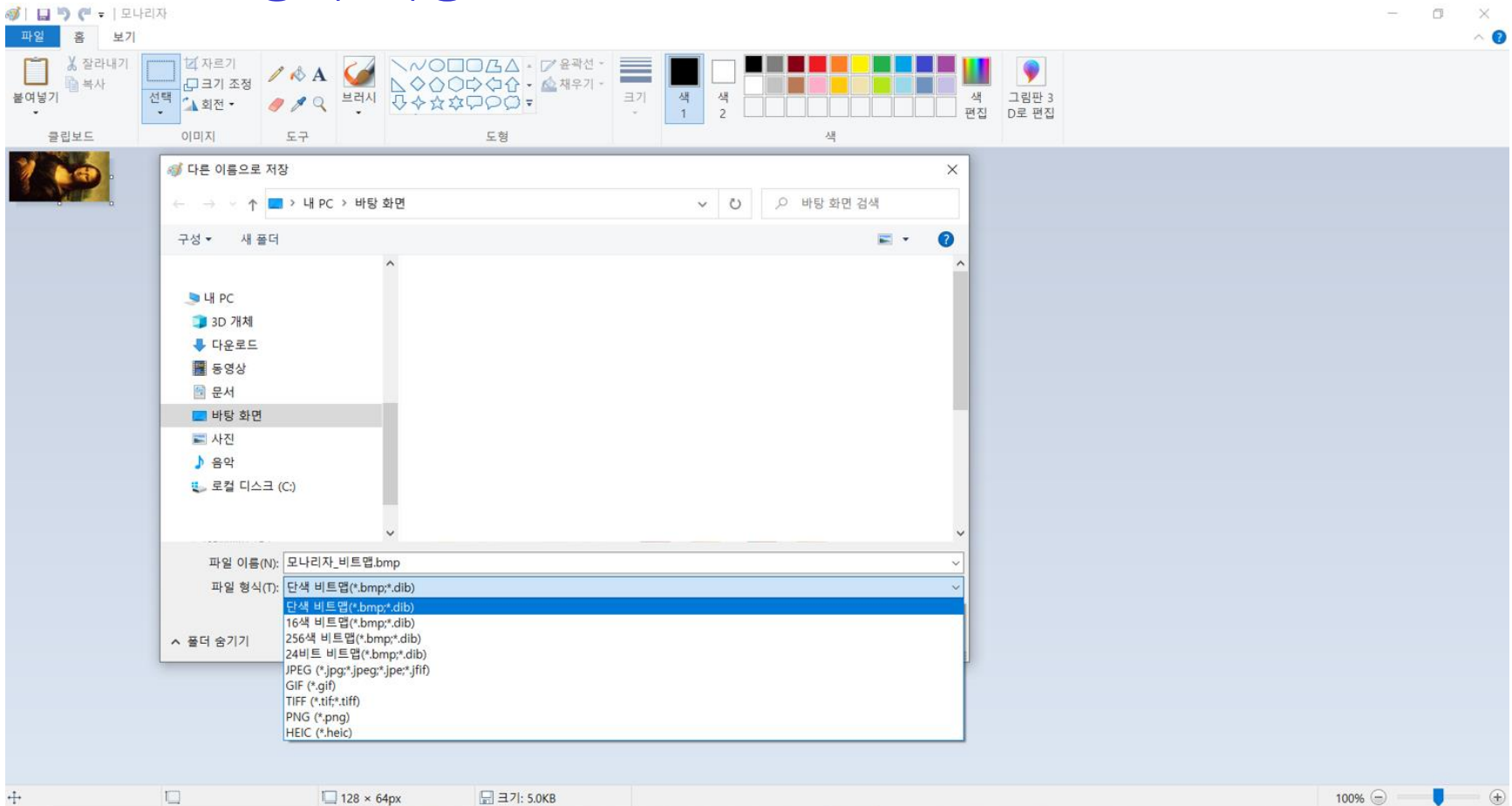
1단계 – 이미지를 비트맵 이미지로 변환하기

❖원하는 이미지를 그림판에 넣고 [크기 조정]-[픽셀]-[가로 세로 비율 유지] 클릭 후 원하는 크기 입력. 단, 가로세로 최대 크기는 128*64



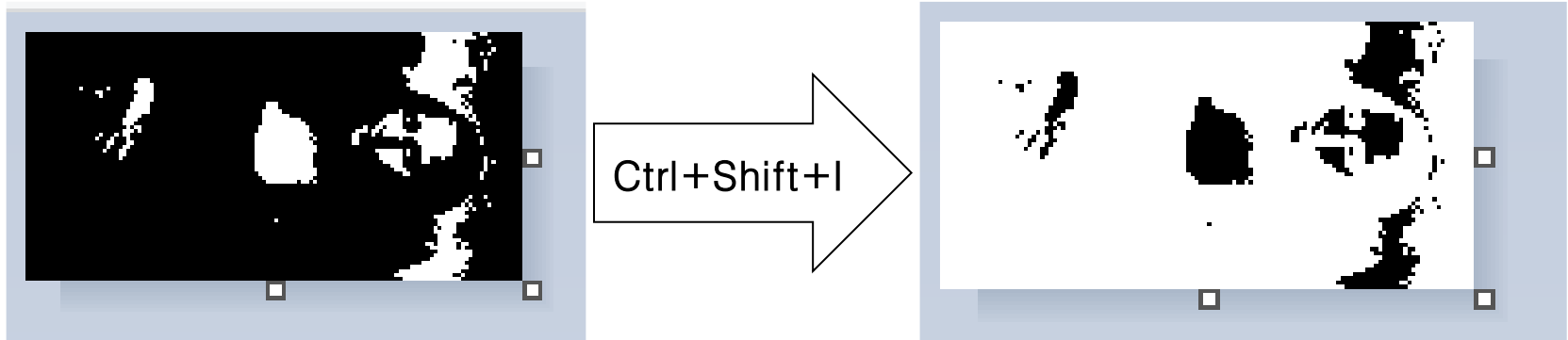
1단계 – 이미지를 비트맵 이미지로 변환하기

❖ [파일] – [다른 이름으로 저장] 클릭 후 파일형식을 “단색 비트맵”으로 변경 후 저장



1단계 - 이미지를 비트맵 이미지로 변환하기

- ❖결과를 확인하면 그림판에서 왼쪽과 같은 결과가 나오지만, OLED모
듈에 출력 될 때 색반전이 일어나기 때문에 그림판에서
[Ctrl+Shift+I] 를 눌러 색을 반전시킨 후 다시 저장한다.



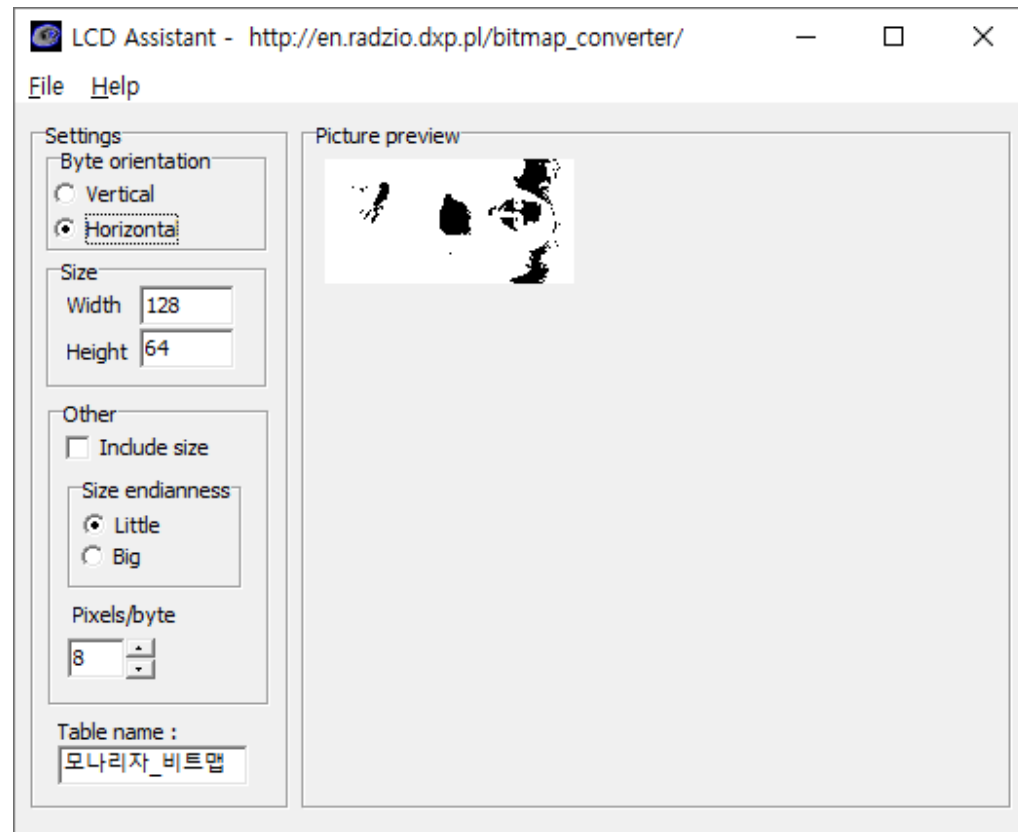
2단계 - 비트맵 코드로 변환하기

❖ http://en.radzio.dxp.pl/bitmap_converter/ 에 접속하여 프로그램을 다운로드하고 압축 해제 후 실행한다.

The screenshot shows the website en.radzio.dxp.pl in a web browser. The page has a dark blue background with white text. On the left, there is a sidebar with links for 'Main menu', 'Tutorials', 'Code library for LCD', and 'ATMEL AVR microcontrollers'. The main content area features a 'LCD Assistant' section with a description of the tool and a 'Download' button. A red box highlights the text 'Download : [LCD Assistant.zip](#)'. Below this, there is a link for 'LCD Assistant CMD.zip'. At the bottom, a small window titled 'LCD Assistant' is shown, displaying settings for 'Byte orientation' (Vertical) and 'Size' (Width: 132, Height: 64).

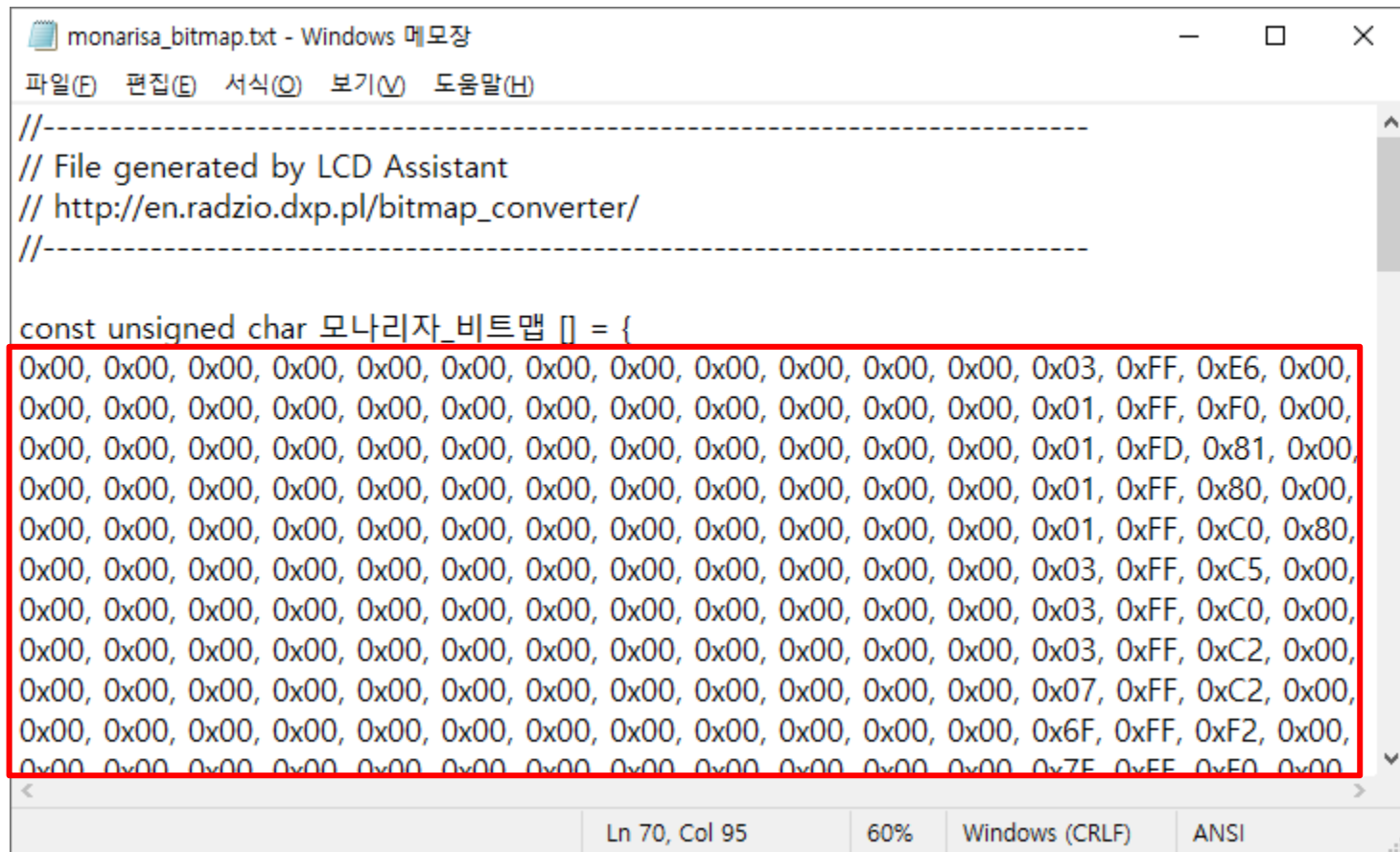
2단계 - 비트맵 코드로 변환하기

- ❖ 프로그램 실행 후 [File]-[Load image] 클릭 후 저장한 사진을 불러온다.
- ❖ [Byte orientation]을 “Vertical” 에서 “Horizontal” 로 변경한다



2단계 - 비트맵 코드로 변환하기

- ❖ [File] - [Save output] 을 클릭하여 저장하고 메모장으로 열어서 확인하면 추출된 코드를 확인할 수 있다.



```
monarisa_bitmap.txt - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

//-----
// File generated by LCD Assistant
// http://en.radzio.dxp.pl/bitmap_converter/
//-----

const unsigned char 모나리자_비트맵 [] = {
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xFF, 0xE6, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0xFF, 0xF0, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0xFD, 0x81, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0xFF, 0x80, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0xFF, 0xC0, 0x80,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xFF, 0xC5, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xFF, 0xC0, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xFF, 0xC2, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xFF, 0xC2, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x6F, 0xFF, 0xF2, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x7E, 0xFF, 0xF0, 0x00,

```

3단계 - 코드 수정

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

#define OLED_RESET 4

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

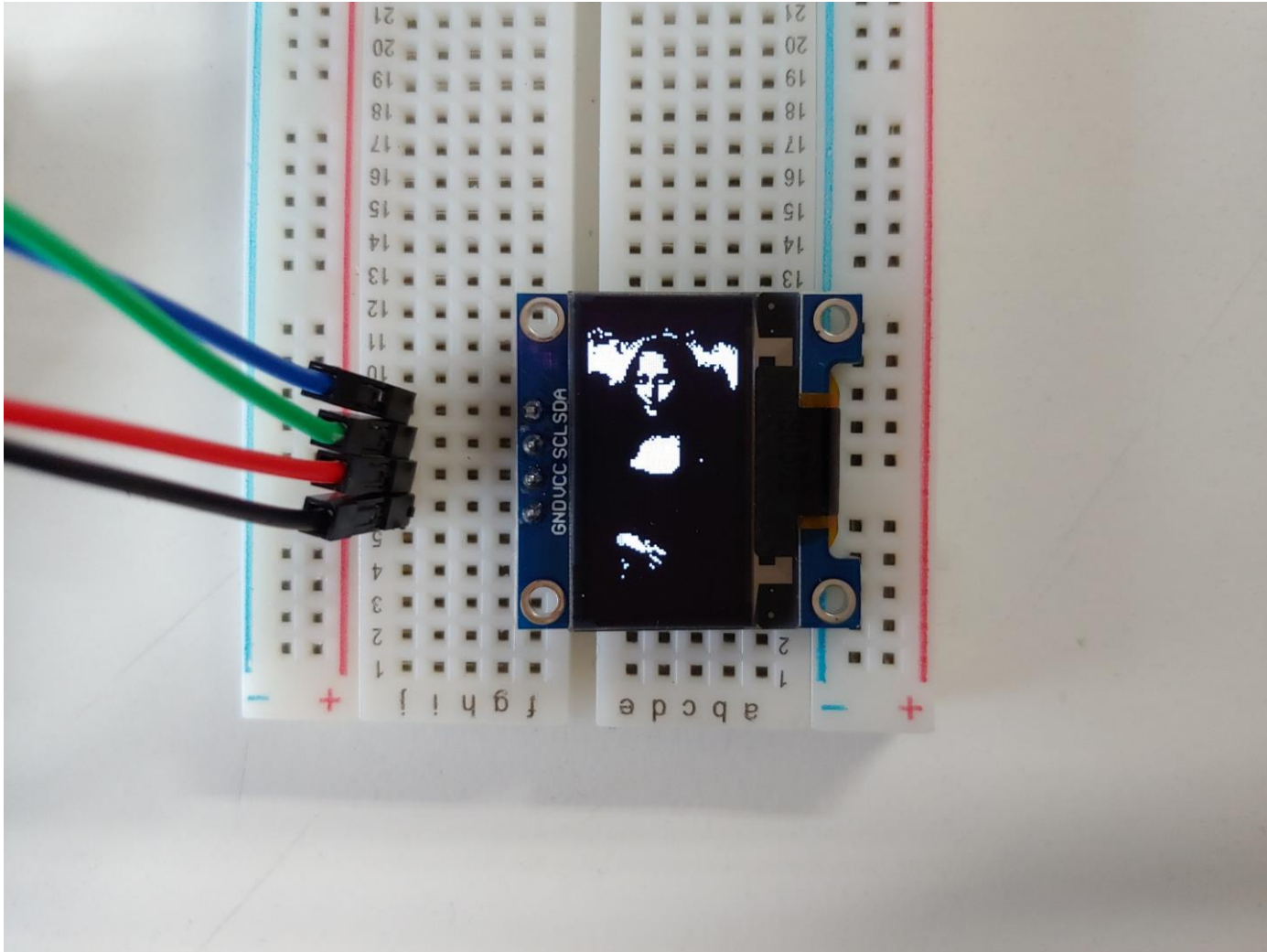
static const unsigned char PROGMEM bmpHex [] = {
  //변환한 비트맵 코드 입력
};

void setup() {
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
}

void loop() {
  display.clearDisplay();
  display.drawBitmap(0,0,bmpHex,128,64,1);
  display.display();
}
```

I2C 주소확인 후 입력

3단계 - 이미지 출력 확인



함수소개

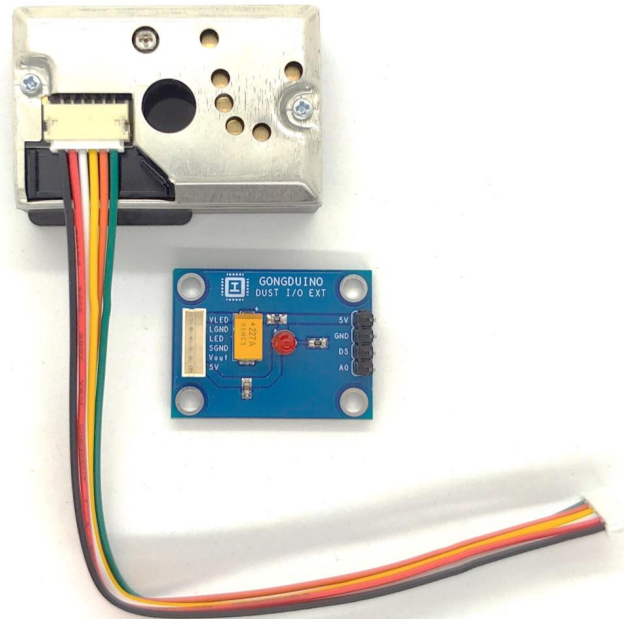
■ `display.drawBitmap(x,y,bitmap,width,height,color);`

- ❖ 이미지를 그리는 함수.
- ❖ x: 이미지를 그리기 시작할 x좌표.
- ❖ Y: 이미지를 그리기 시작할 y좌표.
- ❖ bitmap: 이미지 코드 배열
- ❖ width: 이미지 너비(이미지 너비와 높이는 이미지 크기와 동일해야 한다)
- ❖ height: 이미지 높이(이미지 너비와 높이는 이미지 크기와 동일해야 한다)
- ❖ color: 색상, 기본값 1

3. 미세먼지 센서

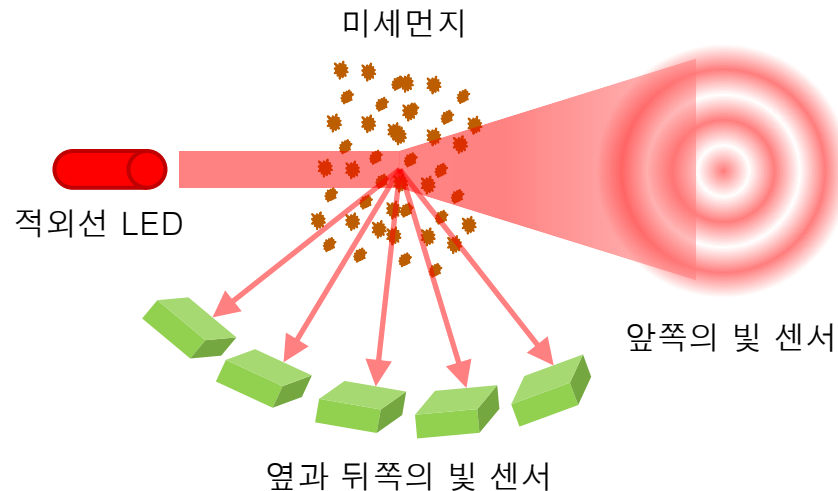
미세먼지 센서 란

- 말 그대로 미세먼지를 측정하는 센서.
- 산란되는 빛을 측정하는 원리로 동작한다.



미세먼지 센서 동작 원리

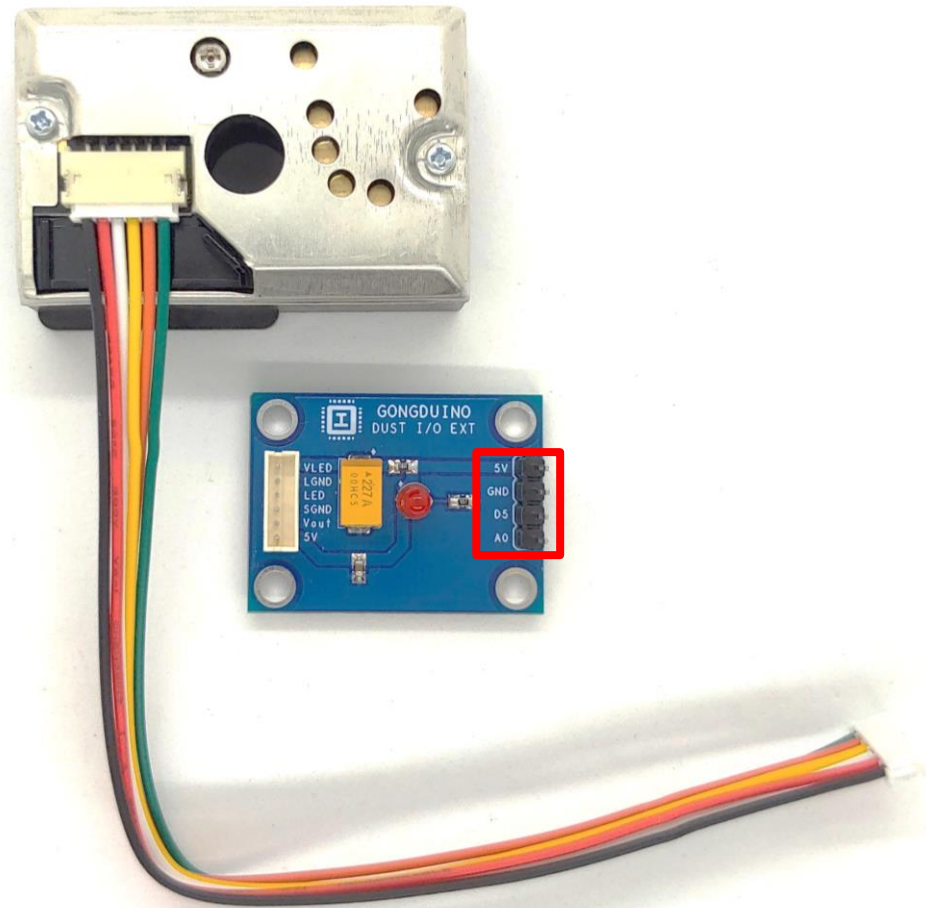
- ❖우리들이 사용하는 간이 측정기는 광산란 방식을 이용하며 짧은 시간의 평균을 알려준다.
- ❖일반적으로 입자는 빛에 노출되면 산란, 굴절, 반사, 흡수 등의 다양한 광학적 특성을 나타낸다.
- ❖입자가 작으면 빛이 많이 산란되고 입자가 크면 빛이 앞에 집중된다.
- ❖먼지가 지나는 곳에 레이저 불빛을 비추고 광학센서는 회절, 굴절, 반사되는 정도를 센싱하여 어느 정도 크기의 입자가 몇개 존재하는지 세어서 농도를 계산한다.



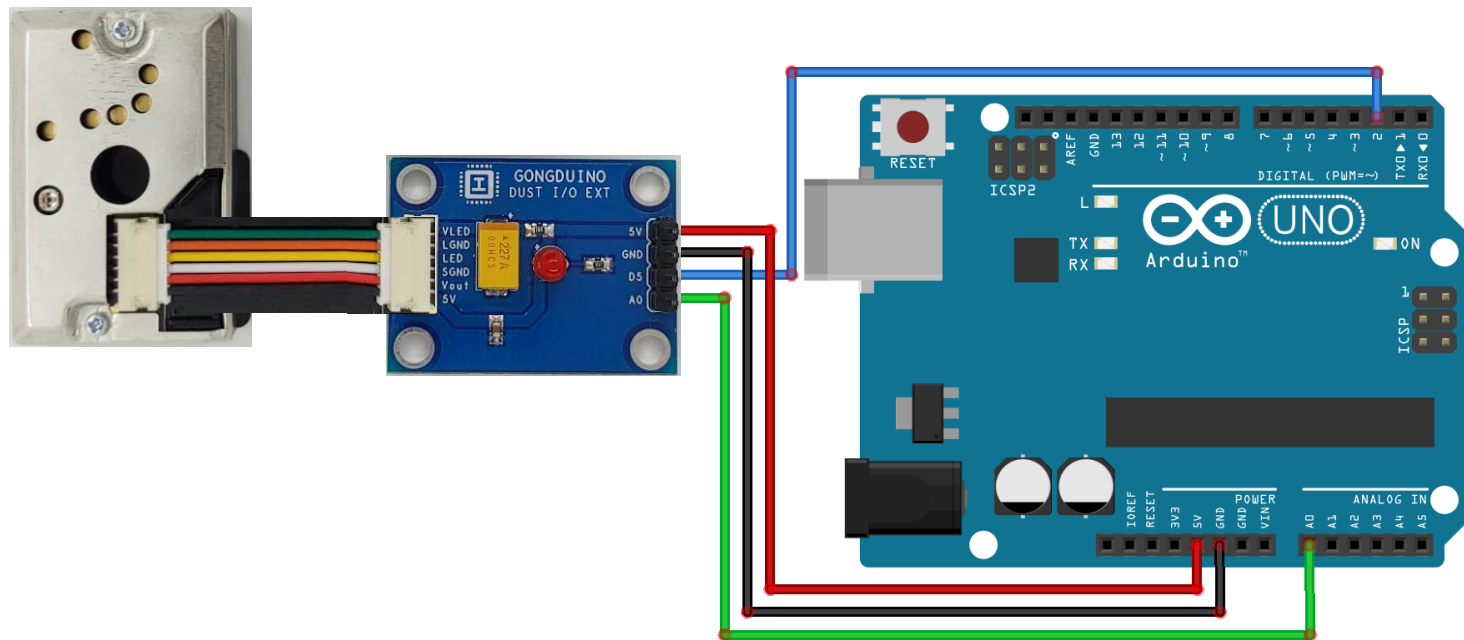
미세먼지 센서 모듈 핀 아웃

■ 다음과 같이 연결

- ❖ 5V – 5V
- ❖ GND – GND
- ❖ D5 – D2
- ❖ A0 – A0



연결도



예제 코드

```
#define MEASURE_PIN 0
#define INFRARED_PIN 2

float measure_value = 0;
float voltage = 0;

void setup(){
  Serial.begin(9600);
  pinMode(INFRARED_PIN, OUTPUT);
  pinMode(MEASURE_PIN, INPUT);
}

void loop(){
  digitalWrite(INFRARED_PIN, LOW);
  delayMicroseconds(280);
  measure_value = analogRead(MEASURE_PIN);
  delayMicroseconds(40);
  digitalWrite(INFRARED_PIN, HIGH);
  delayMicroseconds(9680);

  voltage = measure_value * 5.0 / 1024.0;

  Serial.print("Voltage: ");
  Serial.println(voltage);

  delay(1000);
}
```

동작 확인

COM5

전송

```

14:43:16.708 -> Voltage: 0.75
14:43:17.733 -> Voltage: 0.80
14:43:18.731 -> Voltage: 0.79
14:43:19.762 -> Voltage: 0.80
14:43:20.763 -> Voltage: 0.82
14:43:21.793 -> Voltage: 0.84
14:43:22.817 -> Voltage: 0.77
14:43:23.818 -> Voltage: 0.79
14:43:24.823 -> Voltage: 0.82
14:43:25.852 -> Voltage: 0.80
14:43:26.886 -> Voltage: 0.80
14:43:27.885 -> Voltage: 0.84
14:43:28.917 -> Voltage: 0.83
14:43:29.918 -> Voltage: 0.86
14:43:30.917 -> Voltage: 0.88
14:43:31.951 -> Voltage: 0.86
14:43:32.950 -> Voltage: 0.86
14:43:33.979 -> Voltage: 0.87
14:43:34.988 -> Voltage: 0.93
14:43:36.023 -> Voltage: 0.96
14:43:37.025 -> Voltage: 0.90
14:43:38.055 -> Voltage: 0.92
14:43:39.047 -> Voltage: 0.81
14:43:40.087 -> Voltage: 0.79
14:43:41.084 -> Voltage: 0.81
14:43:42.087 -> Voltage: 0.72
14:43:43.119 -> Voltage: 0.81

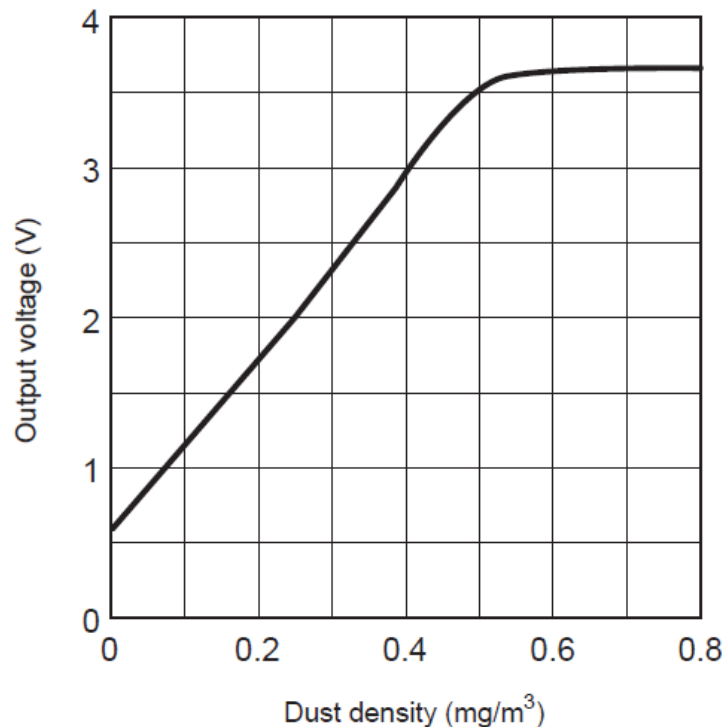
```

☒ 자동 스크롤
 ☒ 타임스탬프 표시
 새 줄
 9600 보드레이트
 출력 지우기

전압 값을 미세먼지 값으로 변환하기

■데이터시트를 보면 센서에서의 출력 전압 값과 미세먼지 농도의 관계 확인할 수 있는 그래프가 있다.

Fig. 3 Output Voltage vs. Dust Density



전압 값을 미세먼지 값으로 변환하기

■ 하지만 미세먼지가 없을 때, 전압 값은 항상 정해져 있지 않기 때문에 초기값이 달라진다. 최대한 정밀한 측정을 위해서 미세먼지가 없을 때 전압 값을 구해야 한다.

- 공기청정기를 이용하면 효과적이다.

■ 초기값을 찾는 가장 좋은 방법은 공기청정기 위에 센서를 올려 두고 나오는 값들의 평균을 구하는 방법이다.

■ 정상적인 데이터 유무는 공개되는 주변 미세먼지 농도와 비교하여 판별이 가능하다.

예제 코드

```
#define MEASURE_PIN 0
#define INFRARED_PIN 2

float measure_value = 0;
float voltage = 0;
float dustDensity = 0;

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

```
void loop(){
  digitalWrite(INFRARED_PIN, LOW);
  delayMicroseconds(280);
  measure_value = analogRead(MEASURE_PIN);
  delayMicroseconds(40);
  digitalWrite(INFRARED_PIN, HIGH);
  delayMicroseconds(9680);

  voltage = measure_value * 5.0 / 1024.0;
  dustDensity = (voltage - 0.7) / 0.005;

  Serial.print("Voltage: ");
  Serial.println(voltage);
  Serial.print("Dust Density: ");
  Serial.println(dustDensity);

  Serial.println();

  delay(1000);
}
```

초기값
보정

동작 확인

COM5

전송

```

14:38:31.736 -> Voltage: 0.83
14:38:31.736 -> Dust Density: 26.02
14:38:31.736 ->
14:38:32.736 -> Voltage: 0.80
14:38:32.736 -> Dust Density: 19.18
14:38:32.736 ->
14:38:33.769 -> Voltage: 0.78
14:38:33.769 -> Dust Density: 15.27
14:38:33.769 ->
14:38:34.776 -> Voltage: 0.80
14:38:34.776 -> Dust Density: 19.18
14:38:34.776 ->
14:38:35.804 -> Voltage: 0.81
14:38:35.804 -> Dust Density: 21.13
14:38:35.804 ->
14:38:36.800 -> Voltage: 0.80
14:38:36.800 -> Dust Density: 20.16
14:38:36.800 ->
14:38:37.833 -> Voltage: 0.81
14:38:37.833 -> Dust Density: 21.13
14:38:37.833 ->
14:38:38.833 -> Voltage: 0.83
14:38:38.833 -> Dust Density: 26.02
14:38:38.833 ->
14:38:39.862 -> Voltage: 0.80
14:38:39.862 -> Dust Density: 20.16
14:38:39.862 ->

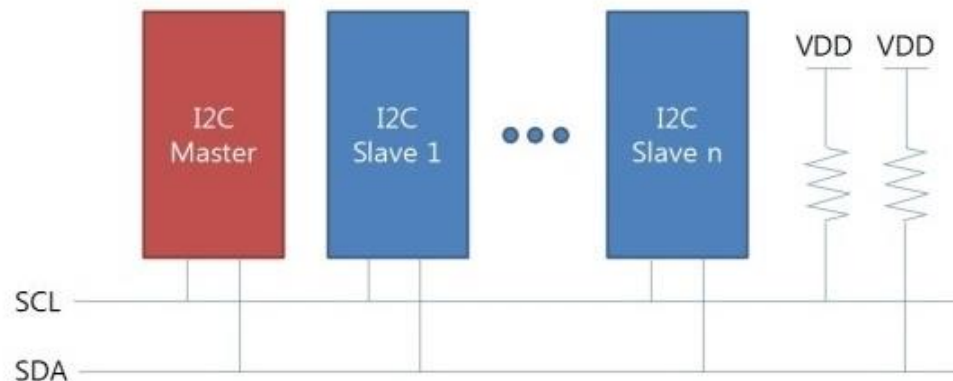
```

☒ 자동 스크롤
 ☒ 타임스탬프 표시
 새 줄
 9600 보드레이트
 출력 지우기

I2C 통신의 이해

I2C 란

- Inter Integrated Circuit
- 마이크로프로세서와 저속 주변장치 사이의 통신을 용도로 Philips에서 개발
- SCL, SDA 두 가닥의 라인을 이용
- Master-Slave 형태로 동작
- 여러 디바이스 연결 시 I2C 라인에 병렬로 연결



예제 코드 – I2C SCANNER 코드

```
#include <Wire.h>

void setup(){
  Wire.begin();
  Serial.begin(9600);
  Serial.println("I2C Scanner");
}

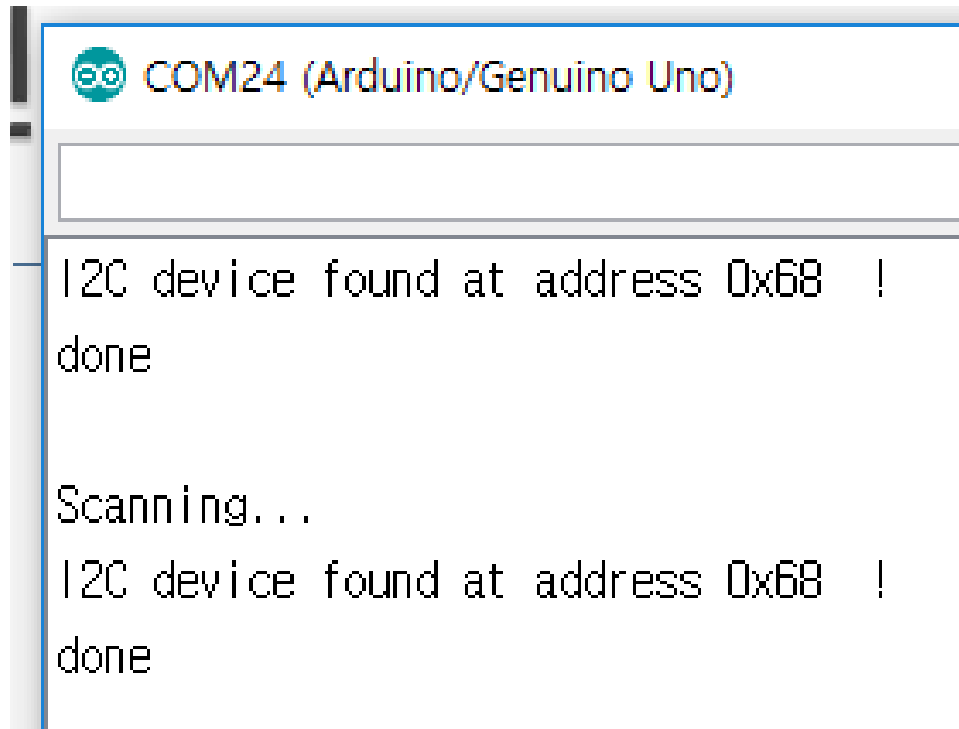
void loop(){
  byte error, address;
  int nDevices;
  Serial.println("Scanning...");
  nDevices = 0;

  for(address = 1; address < 127; 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);
}
```

I2C SCANNER – 확인

- 업로드 후 I2C 시리얼 모니터를 연 뒤 주소 확인
- 주소를 잘 기억해둔다.



```
COM24 (Arduino/Genuino Uno)

I2C device found at address 0x68 !
done

Scanning...
I2C device found at address 0x68 !
done
```