# 사물인터넷

# Chapter 6. LCD 패널

1. LCD 표시장치
2. Liquid Crystal I2C 라이브러리 설치
3. 아두이노 I2C LCD의 제어
4. 사용자 정의 문자열의 출력
5. 센서 값을 I2C LCD1602 표시장치에 출력

# 학습목표

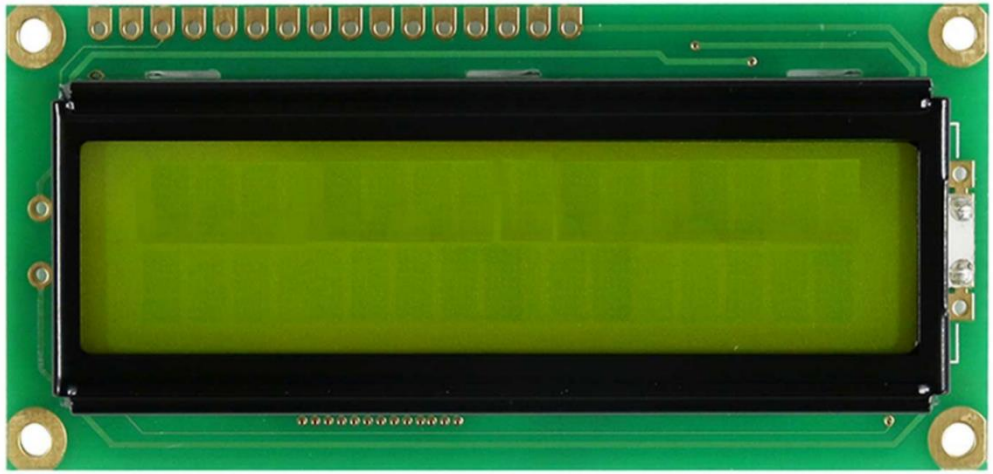
- 6×2 Character LCD의 16개 핀의 기능을 이해한다.
- I2C LCD 모듈의 역할을 이해한다.
- 아두이노 보드와 I2C LCD1602의 회선을 바르게 연결할 수 있다.
- Liquid Crystal I2C 라이브러리를 설치할 수 있다.
- LCD 표시장치의 행과 열의 좌표를 이해하고 적절한 위치에 원하는 문자를 출력할 수 있다.

# 1. LCD 표시장치

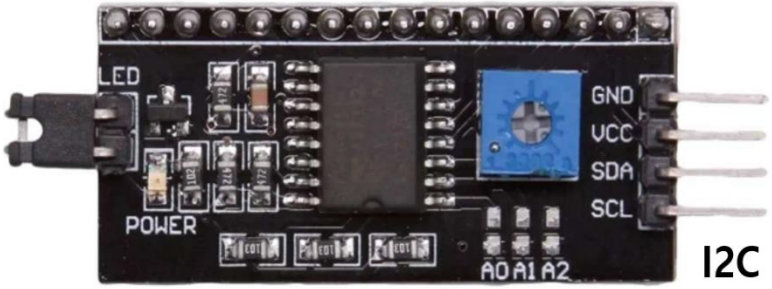
- 액체처럼 유체의 성질을 가지면서 동시에 고체와 같이 광학적 성질을 가지는 액정이라는 물질의 전기적 성질을 이용하여 문자를 표시해주는 장치
- 스스로 빛을 내지 않기 때문에 대부분의 LCD는 후면에 백라이트를 두고, 전면에 액정을 두어 액정이 전기 신호에 따라 빛을 차단하거나 통과시키는 방식
- LCD는 시야 각도가 제한적이고 어두운 곳에서는 선명도가 떨어진다는 단점이 있지만 전력 소모가 적어서 임베디드 시스템에 많이 활용됨

# 1. LCD 표시장치

- I2C LCD1602 표시장치



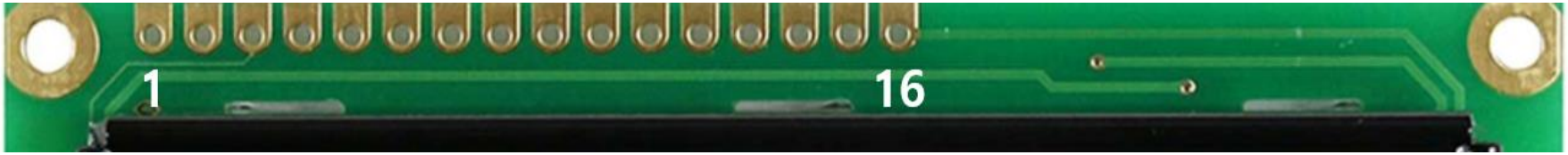
16×2 Character LCD




I2C LCD 모듈

# 1. LCD 표시장치

- 16×2 Character LCD 표시장치




- LCD 표시장치의 외부 인터페이스 신호

표 6.1 LCD 표시장치의 외부 인터페이스 신호

| 핀 번호 | 명칭 | 기능 |
|---|---|---|
| 1 | $V_{SS}$ | Ground |
| 2 | $V_{DD}$ | 전원 |
| 3 | $V_O$ | LCD 밝기 조절 |
| 4 | RS | 레지스터 선택 |
| 5 | R/W | 읽기/쓰기 |

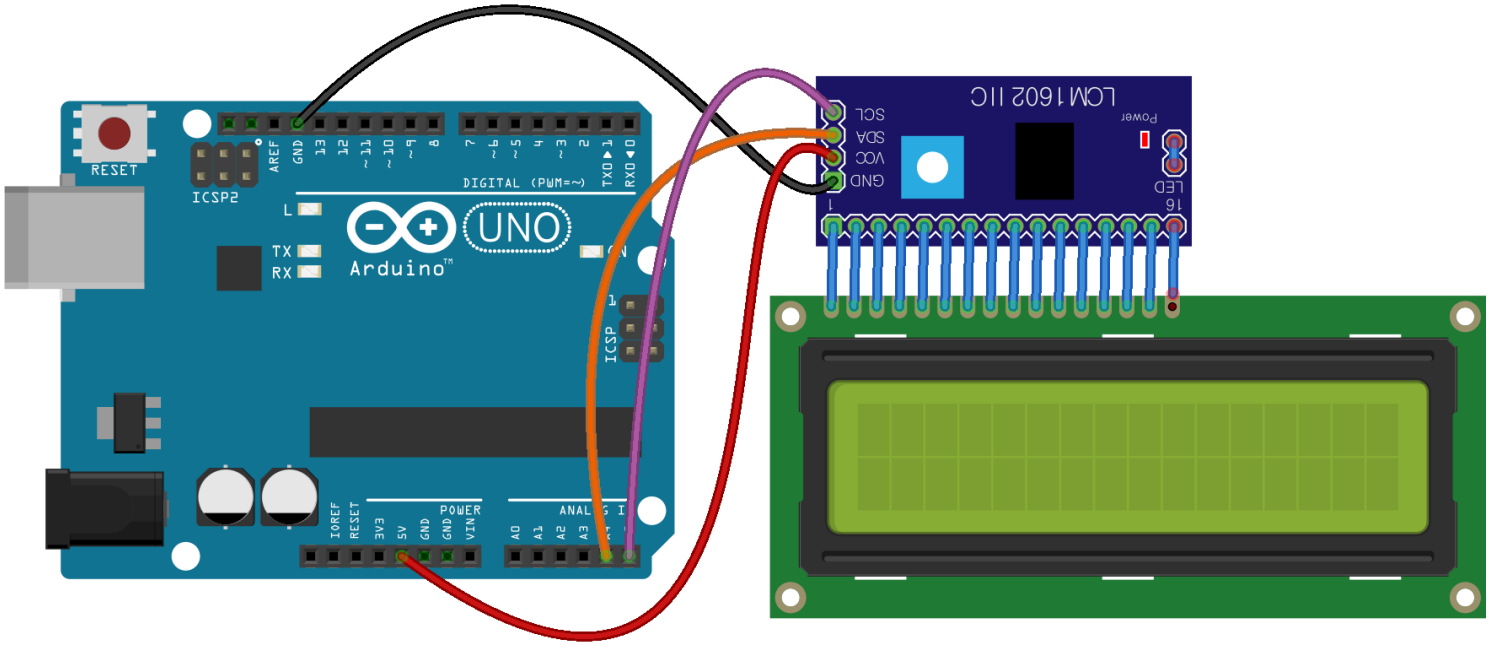
| 핀 번호 | 명칭 | 기능 |
|---|---|---|
| 6 | E | 읽기/쓰기 가능 신호 |
| 7~14 | D0 ~ D7 | 데이터 입출력 |
| 15 | BLA | 백라이트 애노드 |
| 16 | BLK | 백라이트 캐소드 |
| | | |

# 1. LCD 표시장치

- 아두이노 보드와 I2C LCD1602의 회선 연결

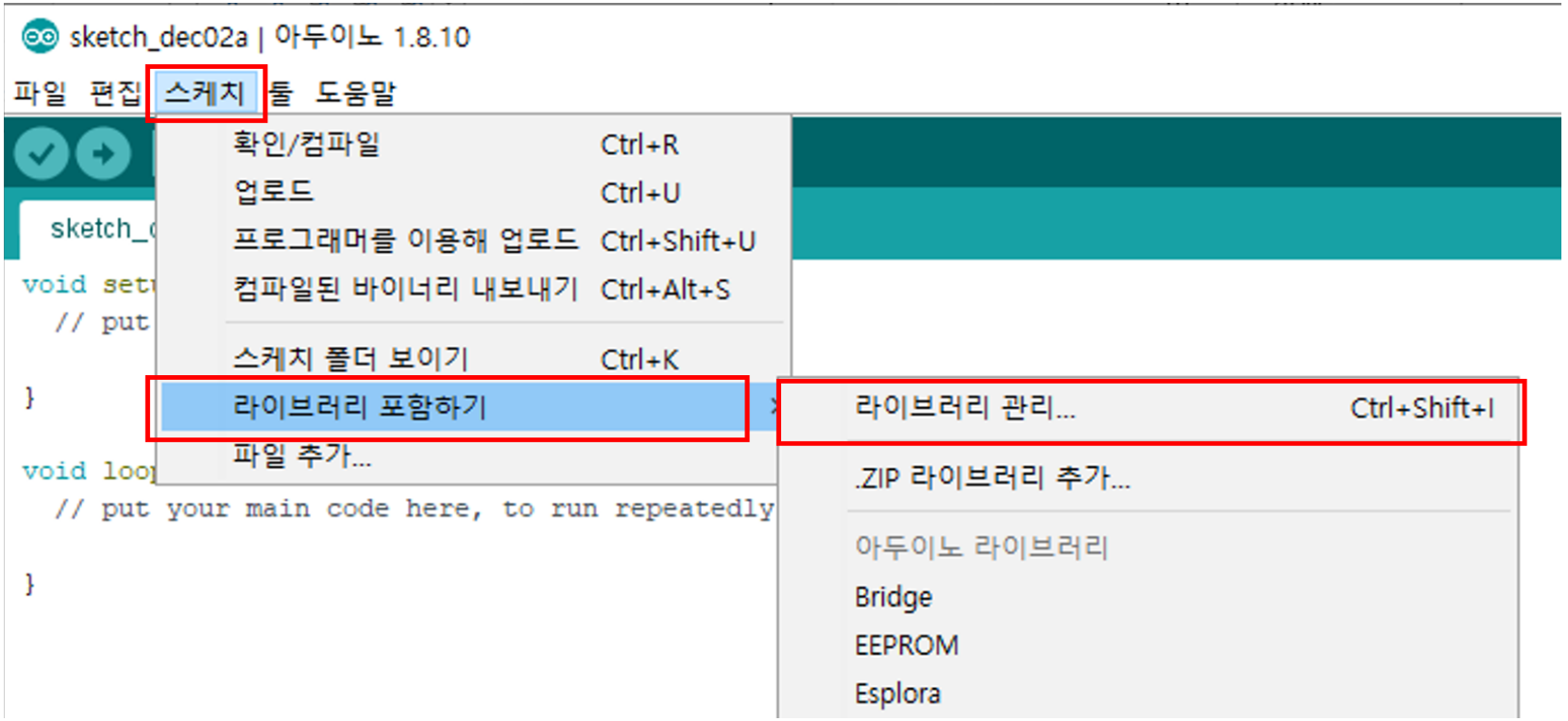
표 6.2 아두이노 보드와 I2C LCD1602의 회선 연결

| I2C LCD1602 | GND | VCC | SDA | SCL |
|---|---|---|---|---|
| 아두이노 우노 | GND | 5V | A4 | A5 |



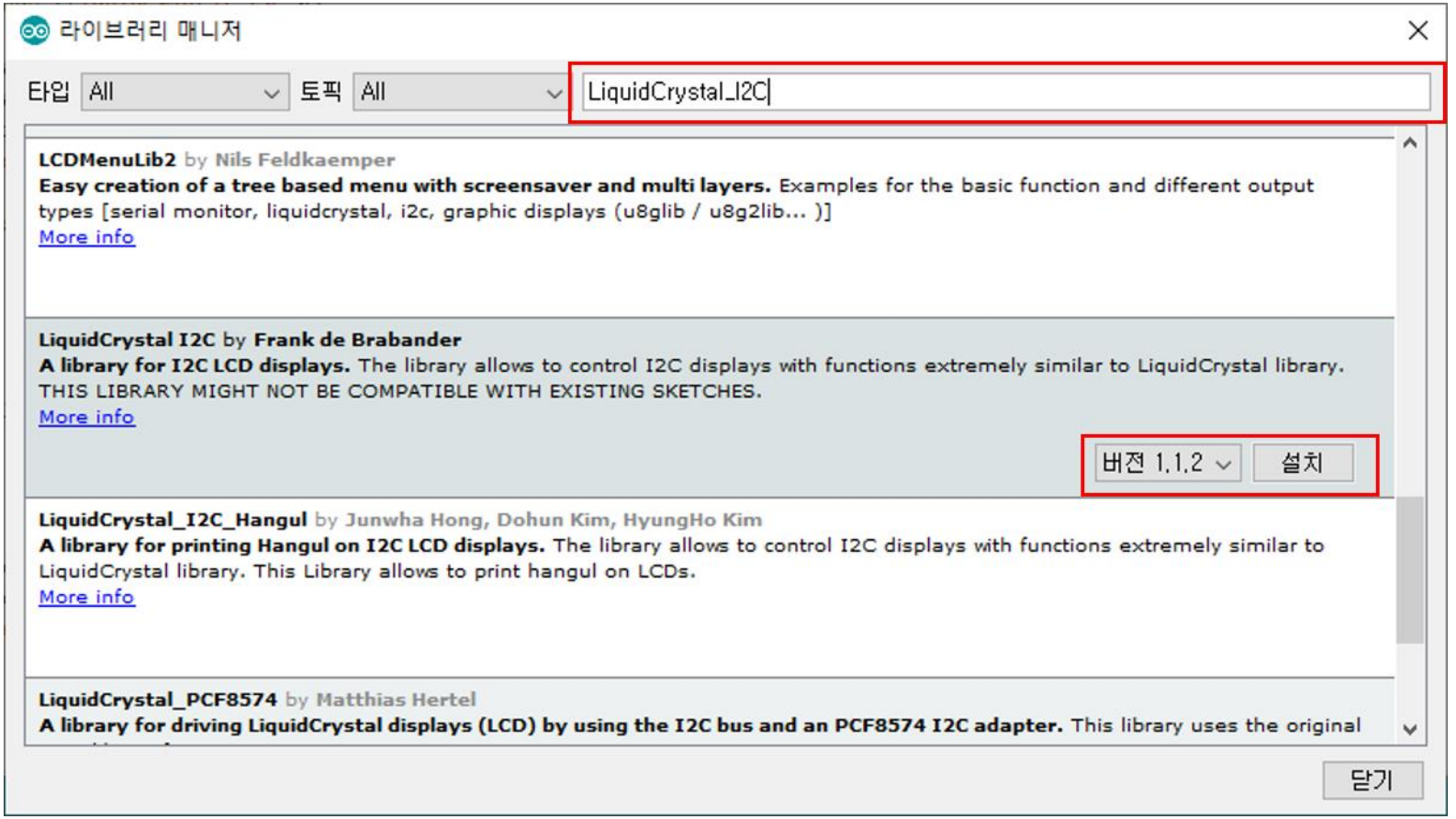
# 2. Liquid Crystal I2C 라이브러리 설치

- 아두이노 IDLE에서 [스케치]→[라이브러리 포함하기]→[라이브러리 관리...]를 실행



# 2. Liquid Crystal I2C 라이브러리 설치

- 라이브러리 매니저 창이 나타나면 'LiquidCrystal_I2C'로 검색한 후 Frank de Brabander가 만든 'LiquidCrystal I2C' 라이브러리를 설치



# 3. 아두이노 I2C LCD의 제어

- I2C LCD 모듈의 제어 함수

| | 함수 | 설명 |
|---|---|---|
| 1 | LiquidCrystal_I2C lcd(0x27,16,2); | 0x27는 I2C 주소를 가지고 있는 16x2 LCD객체 생성 |
| 2 | lcd.begin(); | LCD 사용 시작 |
| 3 | lcd.init(); | LCD 초기화 |
| 4 | lcd.backlight(); | LCD 백라이트를 켬 |
| 5 | lcd.noBacklight(); | LCD 백라이트를 끔 |
| 6 | lcd.display(); | LCD에 내용 표시 |
| 7 | lcd.noDisplay(); | LCD의 내용 숨김 |
| 8 | lcd.setCursor(col,row); | row, col의 좌표로 커서를 옮김 |
| 9 | lcd.home(); | 커서를 0,0의 좌표로 이동 |
| 10 | lcd.cursor(); | LCD에 커서 표시 |
| 11 | lcd.noCursor(); | LCD의 커서 숨김 |
| 12 | lcd.blink(); | 커서를 깜빡임 |
| 13 | lcd.noBlink(); | 커서를 깜빡이지 않음 |
| 14 | lcd.print(string); | LCD 화면에 문자열을 출력 |
| 15 | lcd.write(char); | LCD 화면에 문자 출력, 아스키 코드 입력 시에는 아스키 코드에 해당하는 문자를 출력 |
| 16 | lcd.clear(); | LCD 화면의 모든 문자를 지움 |
| 17 | lcd.scrollDisplayRight(); | LCD 화면 내용을 오른쪽으로 한 칸 이동 |
| 18 | lcd.scrollDisplayLeft(); | LCD 화면 내용을 왼쪽으로 한 칸 이동 |
| 19 | lcd.autoscroll(); | LCD 화면 내용을 자동으로 오른쪽에서 왼쪽으로 이동 |

# 3. 아두이노 I2C LCD의 제어

- "Welcome to ARDUINO world !!" 문자 출력

```
#include <LiquidCrystal_I2C.h>
// 16×2 크기의 I2C LCD 객체를 생성
// 16 글자 2줄짜리 디스플레이의 I2C LCD 주소는 주로 0x27로 설정
LiquidCrystal_I2C lcd(0x27, 16, 2);
void setup()
{
  lcd.init(); // I2C LCD의 초기화
  lcd.backlight(); // I2C LCD의 백라이트 켜기
}
void loop()
{
  lcd.setCursor(0, 0); // 커서를 0번 행(윗줄)의 0번 열(가장 왼쪽 열)에 위치
  lcd.print("Welcome to"); // 메시지를 I2C LCD에 표시
  lcd.setCursor(1, 1); // 커서를 1번행(아랫줄)의 1번 열(왼쪽 두 번째 열)에 위치
  lcd.print("ARDUINO world !"); // 메시지를 I2C LCD에 표시
}
```

# 3. 아두이노 I2C LCD의 제어

- 커서 깜빡임 제어

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
void setup()
{
  lcd.init(); // I2C LCD의 초기화
  lcd.backlight(); // I2C LCD 백라이트 ON
}
```

```
void loop()
{
  lcd.print("Cursor Blink"); // "Cursor Blink" 출력
  lcd.cursor();  // 커서 보이기
  lcd.blink();    // 깜빡이기
  delay(3000); // 3초 지연
  lcd.clear();    // LCD화면 지우기
  lcd.print("Cursor Out"); // "Cursor Out" 출력
  lcd.noBlink( ); // 깜박임 정지
  delay(3000);  // 3초 지연
  lcd.clear();     // LCD화면 지우기
  lcd.noCursor( ); // 커서 감추기
  delay(3000);    // 3초 지연
  lcd.clear( );     // LCD화면 지우기
}
```

# 3. 아두이노 I2C LCD의 제어

- 문자열 깜박이기

"Hello, arduino!"를 출력하고 이 문자열이 1초 간격으로 깜박이는 프로그램

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
void setup()
{
  lcd.init(); // I2C LCD의 초기화
  lcd.backlight(); // I2C LCD 백라이트 ON
}
```

```
void loop()
{
  lcd.print("Hello!"); // "Hello!" 출력
  lcd.noDisplay(); // 표시 안함
  delay(1000); // 1초 지연
  lcd.display(); // 표시
  delay(1000); // 1초 지연
  lcd.clear( ); // LCD화면 지우기
}
```

# 3. 아두이노 I2C LCD의 제어

- 문자열의 이동

  I2C LCD1602 표시장치에 두 줄의 문장을 출력하고 한 줄씩 차례대로 오른쪽에서 왼쪽으로 한 칸씩 문자열이 이동하는 배너 프로그램을 작성

```
#include <LiquidCrystal_I2C.h>
char array1[]="Hello, world! "; // I2C LCD의 윗줄에 표시할 문장
char array2[]="Nice to meet you. "; // I2C LCD의 아랫줄에 표시할 문장
int delayTime = 500;
LiquidCrystal_I2C lcd(0x27,16,2); // 16×2 디스플레이의 I2C LCD 주소는 0x27로 설정

void setup()
{
  lcd.init(); // I2C LCD의 초기화
  lcd.backlight(); // I2C LCD의 백라이트 켜기
}
```

# 3. 아두이노 I2C LCD의 제어

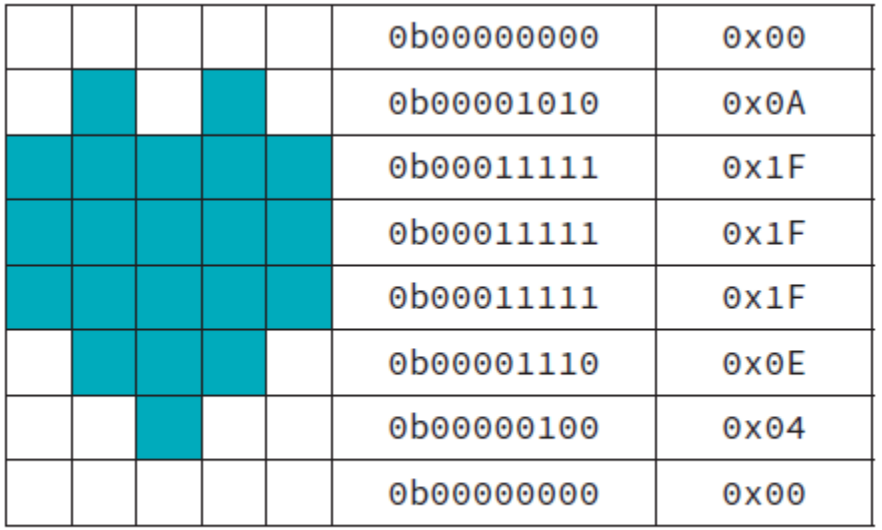
```
void loop()
{
  lcd.setCursor(15,0); // 커서를 0번 행(디스플레이의 윗줄)의 15번 열에 위치
  for (int posCnt1 = 0; posCnt1 < 26; posCnt1++)
  {
    lcd.scrollDisplayLeft(); // 출력되는 문자열을 왼쪽으로 한 칸씩 이동
    lcd.print(array1[posCnt1]); // 메시지를 I2C LCD에 표시
    delay(delayTime);
  }
  lcd.clear(); // I2C LCD 화면을 지우고 커서를 왼쪽 상단에 위치
  lcd.setCursor(15,1); // 커서를 1번 행(디스플레이의 아랫줄)의 15번 열에 위치
  for (int posCnt2 = 0; posCnt2 < 26; posCnt2++)
  {
    lcd.scrollDisplayLeft(); // 출력되는 문자열을 왼쪽으로 한 칸씩 이동
    lcd.print(array2[posCnt2]);
    delay(delayTime);
  }
  lcd.clear();
}
```

# 4. 사용자 정의 문자열의 출력

- 사용자 정의 문자는 5×8 크기

사용자 정의 문자를 생성 : createChar() 함수
LCD 표시장치에 출력 : write(byte(n)) 함수

```
lcd.createChar(n, char_name);
lcd.setCursor(i, j); // j 번째 줄 i 번째 칸으로 이동
lcd.write(byte(n));
```



| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | 0b00000000 | 0x00 |
| | ■ | | ■ | | 0b00001010 | 0x0A |
| ■ | ■ | ■ | ■ | ■ | 0b00011111 | 0x1F |
| ■ | ■ | ■ | ■ | ■ | 0b00011111 | 0x1F |
| ■ | ■ | ■ | ■ | ■ | 0b00011111 | 0x1F |
| | ■ | ■ | ■ | | 0b00001110 | 0x0E |
| | | ■ | | | 0b00000100 | 0x04 |
| | | | | | 0b00000000 | 0x00 |

그림 6.7 하트 문자의 정의

# 4. 사용자 정의 문자열의 출력

- '아두이노♥' 문자 출력

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);
byte char_00[8] = {0x00, 0x0E, 0x1B, 0x11, 0x11, 0x1B, 0x0E, 0x00}; // ㅇ
byte char_01[8] = {0x08, 0x08, 0x08, 0x0E, 0x08, 0x08, 0x08, 0x08}; // ㅏ
byte char_02[8] = {0x1F, 0x10, 0x10, 0x10, 0x10, 0x1F, 0x00, 0x00}; // ㄷ
byte char_12[8] = {0x1F, 0x04, 0x04, 0x04, 0x04, 0X00, 0X00, 0X00}; // ㅜ
byte char_05[8] = {0x04, 0x04, 0x04, 0x04, 0x04, 0x04, 0x04, 0x04}; // ㅣ
byte char_06[8] = {0x00, 0x10, 0x10, 0x10, 0x1E, 0x00, 0x00, 0x00}; // ㄴ
byte char_16[8] = {0x04, 0x04, 0x04, 0x04, 0x1F, 0x00, 0x00, 0x00}; // ㅗ
byte heart[8] = {0x00, 0x0A, 0x1F, 0x1F, 0x1F, 0x0E, 0x04, 0x00}; // 하트
```

# 4. 사용자 정의 문자열의 출력

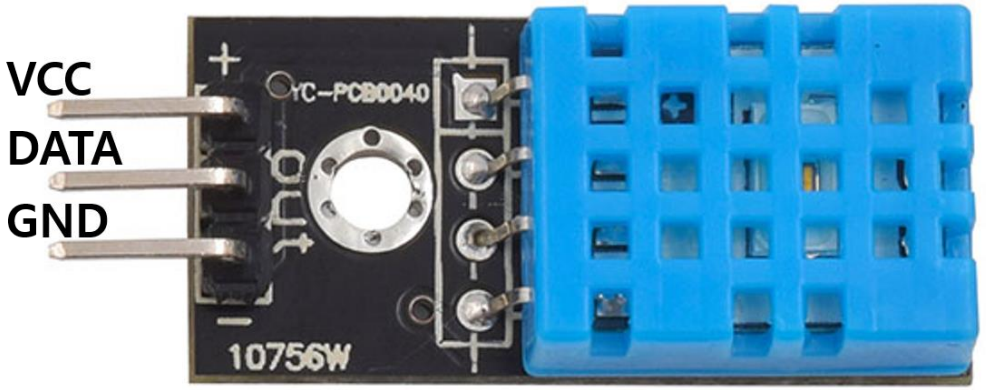
```
void setup()
{
  lcd.init(); // I2C_LCD의 초기화
  lcd.backlight(); // I2C_LCD의 백라이트 켜기
  // LCD 출력 문자 8개까지만 생성 가능
  lcd.createChar(0, char_00); // 정의된 문자 char_00('ㅇ')을 0번 배열에 저장
  lcd.createChar(1, char_01);
  lcd.createChar(2, char_02);
  lcd.createChar(3, char_05);
  lcd.createChar(4, char_06);
  lcd.createChar(5, char_12);
  lcd.createChar(6, char_16);
  lcd.createChar(7, heart);
  lcd.clear();
}
```

# 4. 사용자 정의 문자열의 출력

```
void loop() {
  lcd.setCursor(0, 0); // 첫 번째 줄 첫 번째 칸으로 이동
  lcd.write(byte(7));  // 7번 배열에 저장된 heart 문자 출력
  lcd.setCursor(2, 0);
  lcd.write(byte(0)); // ㅇ
  lcd.setCursor(3, 0);
  lcd.write(byte(1)); // ㅏ
  lcd.setCursor(4, 0);
  lcd.write(byte(2)); // ㄷ
  lcd.setCursor(4, 1);
  lcd.write(byte(5)); // ㅜ
  lcd.setCursor(6, 0);
  lcd.write(byte(0)); // ㅇ
  lcd.setCursor(7, 0);
  lcd.write(byte(3)); // ㅣ
  lcd.setCursor(8, 0);
  lcd.write(byte(4)); // ㄴ
  lcd.setCursor(8, 1);
  lcd.write(byte(6)); // ㅗ
}
```

# 5. 센서의 값 I2C LCD1602 표시장치에 출력

- DHT11 온습도 모듈이 측정한 값 출력
- DHT11
  - 써미스터와 정전식 습도 센서가 내장되어 있는 온습도 센서
  - 써미스터는 온도의 변화에 따라 저항값이 변하는 소자이며, 정전식 습도 센서는 습도의 변화에 따라 저항 값이 변하는 소자

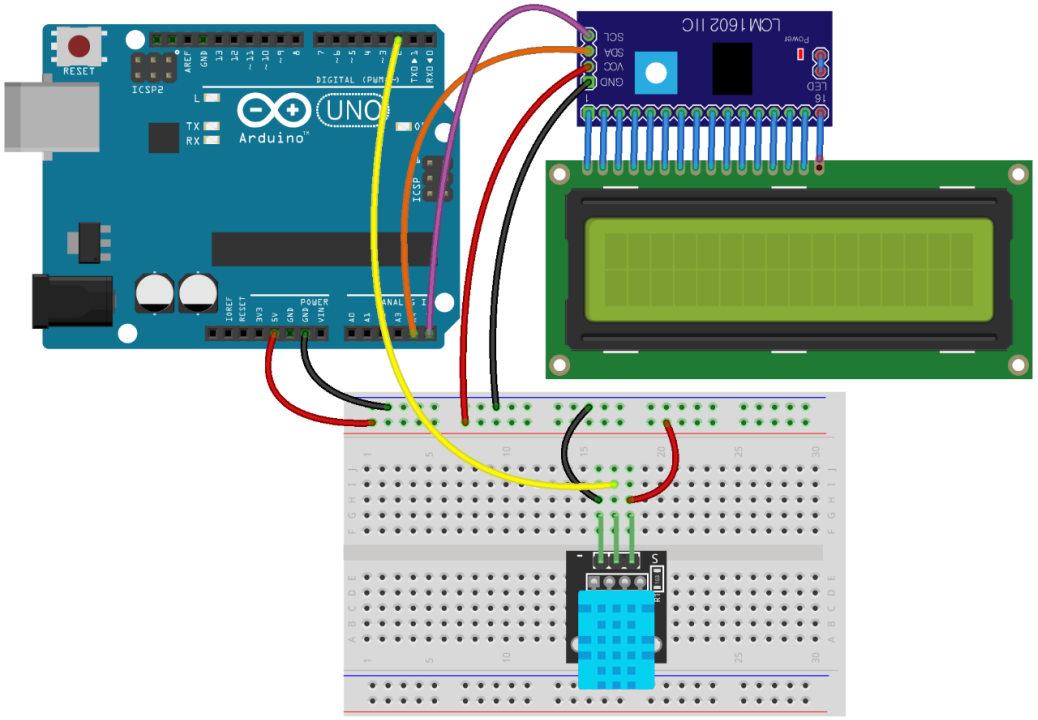


# 5. 센서 값을 I2C LCD1602 표시장치에 출력

- DHT11 모듈과 아두이노 보드의 회로 연결

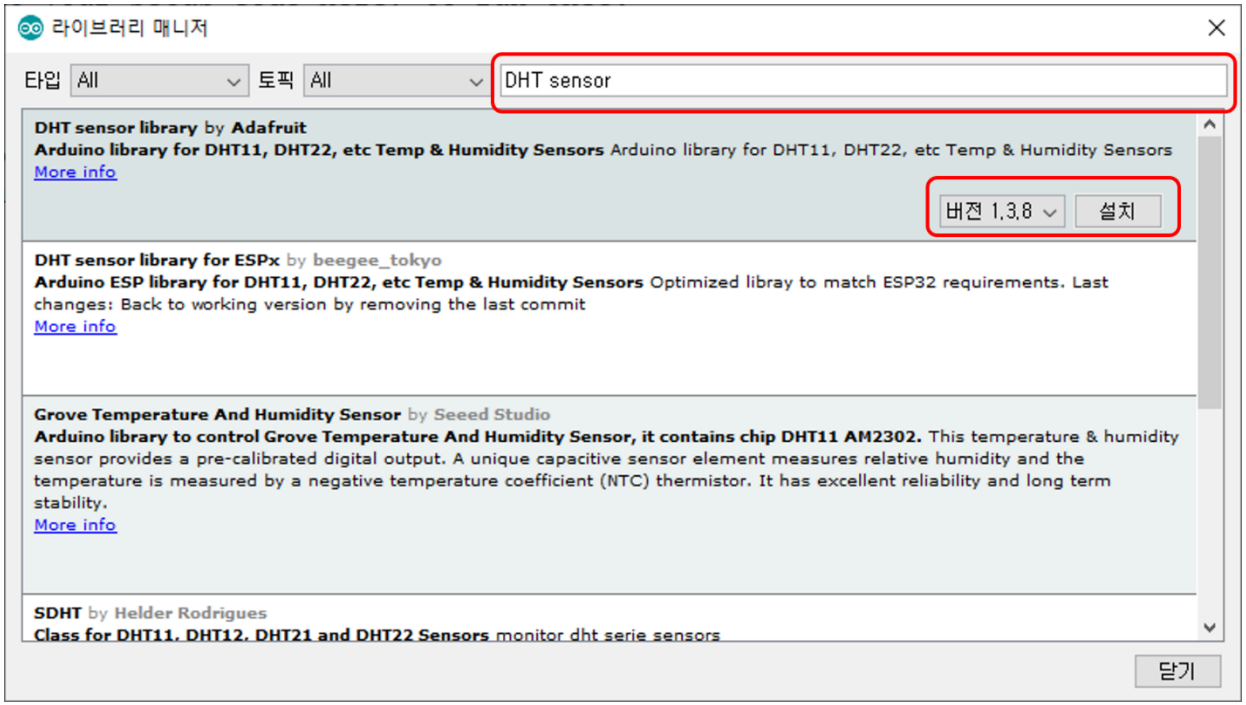
표 6.4 DHT11 모듈과 아두이노 보드의 회로 연결

| DHT11 모듈 | VCC | DATA | GND |
|---|---|---|---|
| 아두이노 보드 | 5V | 디지털 2번 핀 | GND |



# 5. 센서의 값 I2C LCD1602 표시장치에 출력

- DHT 센서 라이브러리 설치
  - 아두이노 IDE의 [툴] 메뉴에서 [라이브러리 관리...]를 클릭
  - "DHT sensor"를 입력하여 나타난 라이브러리 중에서 DHT Sensor Library by Adafruit의 최신 버전을 선택



# 5. 센서의 값 I2C LCD1602 표시장치에 출력

```
#include <LiquidCrystal_I2C.h>
#include <DHT.h>
#define DHTTYPE DHT11 // DHT11
#define DHTPIN 2
DHT dht(DHTPIN, DHTTYPE);
LiquidCrystal_I2C lcd(0x27, 16, 2);
void setup() {
  Serial.begin(9600);
  Serial.println(F("DHTxx test!"));
  dht.begin();
  lcd.init(); // LCD의 초기화
  lcd.backlight(); // 백라이트 오픈
  lcd.clear();
}
```

# 5. 센서의 값 I2C LCD1602 표시장치에 출력

```
void loop() {
  delay(3000);
  float tem = dht.readTemperature();
  float hum = dht.readHumidity();
  // 센서값 읽기 실패 시 프로그램 종료
  if (isnan(tem) || isnan(hum)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }
  // 온도와 습도 값을 LCD 모듈에 출력
  lcd.setCursor(1,0);  lcd.print("Temp : ");
  lcd.setCursor(8,0);  lcd.print(tem, 1);   lcd.print(" C");
  lcd.setCursor(1,1);  lcd.print("Humi : ");
  lcd.setCursor(8,1);  lcd.print(hum, 1);  lcd.print(" %");
  // 온도와 습도 값을 시리얼 모니터에 출력
  Serial.print("Temperature : "); Serial.print(tem);
  Serial.print("  ");
  Serial.print("Humidity : "); Serial.print(hum);
  Serial.println("%");
}
```