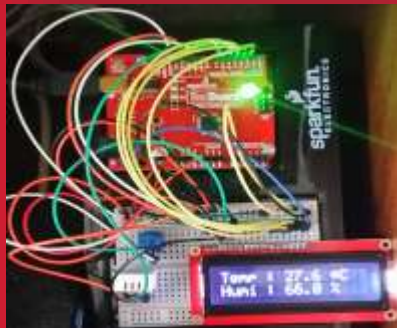




# Arduino-basic

## [wk10]

## Analog Input I.



Learn how to code Arduino from scratch

Comsi, INJE University

1<sup>st</sup> semester, 2022

Email : [chaos21c@gmail.com](mailto:chaos21c@gmail.com)



# My ID (ARnn, github repo)

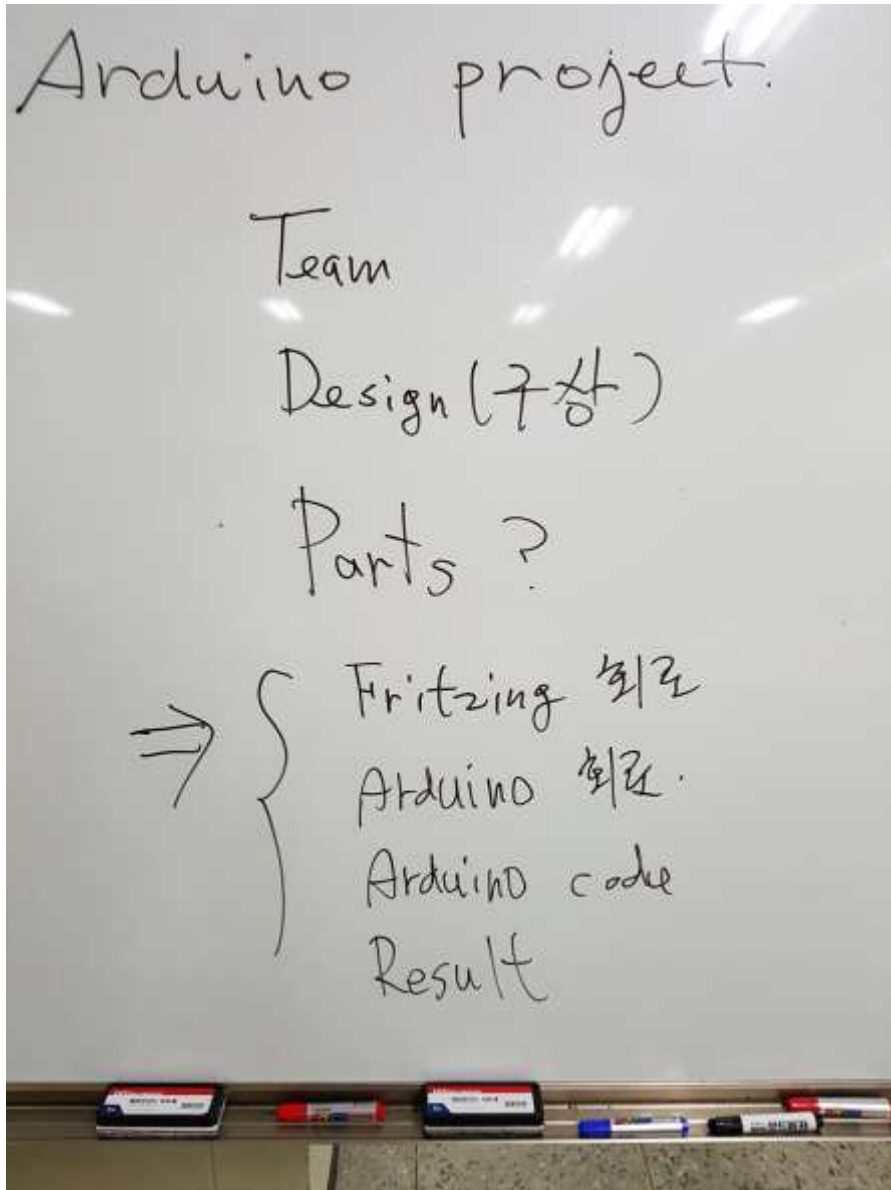
- [AR01 김정헌]
- [AR02 유석진]
- [AR03 김기덕]
- [AR04 강대진]
- [AR05 김성우]
- [AR06 김창연]
- [AR07 김창욱]
- [AR08 김태화]
- [AR09 박세훈]
- [AR10 박신영]

- [AR11 박제홍]
- [AR12 이승무]
- [AR13 이승준]
- [AR14 이재하]
- [AR15 이준희]
- [AR16 이현준]
- [AR17 임태형]
- [AR18 정동현]
- [AR19 정희서]
- [AR20 이한글]
- [AR21 황혁준]
- [AR22 김동영]

# AR Project 2022

Team		발표 제목	기타
1	AR07,09 김성우 박세훈		
2	AR06,10 김찬영 박신영		
3	AR13,15 이승준 이준희		
4	AR12,17 이승무 임태형		
5	AR20,21 이한글 황혁준		
6	AR11,16 박제홍 이현준		
7	AR04,14 강대진 이태하		
8	AR08,18 김태화 정동현		
9	AR01,07 김정현 김창욱		
10			
11			

# Arduino team project



- 2명/팀
- 구상 소개 (5.26, 6.2), ppt 준비
- 부품은 수업 세트 기준
- 팀당 발표 자료 준비
- 발표 : 6월16일 (목)
  - ✓ PPT 발표 및 시연 (동영상도 가능)
- 참고
  - 추가 부품은 조별로 개별 조달.



# [Practice]

## ◆ [wk09]

- **Arduino : Digital input**
- **Complete your project**
- **Submit file : ARnn\_Rpt08**

# wk09 : Practice-08 : ARnn\_Rpt08

## ◆ [Target of this week]

- Complete your works
- Save your outcomes
- Upload all in github.

제출폴더명 : **ARnn\_Rpt08**

### 제출할 파일들

- ① **ARnn\_Switch.ino**
- ② **ARnn\_Switch\_good.ino**
- ③ **ARnn\_Switch\_time.png**
- ④ **ARnn\_Switch\_time.ino**
- ⑤ **ARnn\_all\_keys.png**
- ⑥ **\*.ino**

# wk09 : Practice-08 : ARnn\_Switch\_time

ARnn\_Switch\_time

```

4  */
5  #include <Wire.h>
6  // I2C LCD 라리브러리 설정
7  #include <LiquidCrystal_I2C.h>
8  // LCD I2C address 설정
9  // PCF8574:0x27, PCF8574A:0x3F
10 LiquidCrystal_I2C lcd(0x27,16,2);
11 // LCD address:0x27, 16X2 LCD, 0x3F
12
13 // 2번핀을 스위치 입력으로 설정
14 const int inputPin = 2;
15
16 // 현재의 시간을 저장하기 위한 변수
17 long startTime = 0;
18 // 실제 스위치가 눌린 후 지연되는 시간
19 long swCountTimer = 0;
20
21 void setup() {
22
23     // 스위치 입력을 위하여 2번핀을 입력으로 설정하고 풀업시킨다
24     pinMode(inputPin, INPUT_PULLUP);
25     // 시리얼 통신을 설정한다
26     Serial.begin(9600);
27     lcd.init();
28     lcd.clear();
29     lcd.backlight();
30     lcd.setCursor(0,0);
31     lcd.print("Press button");
32 }

```



```

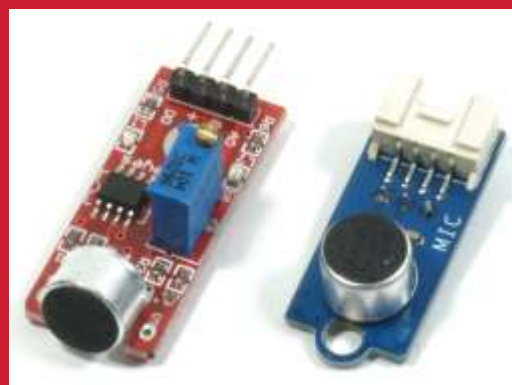
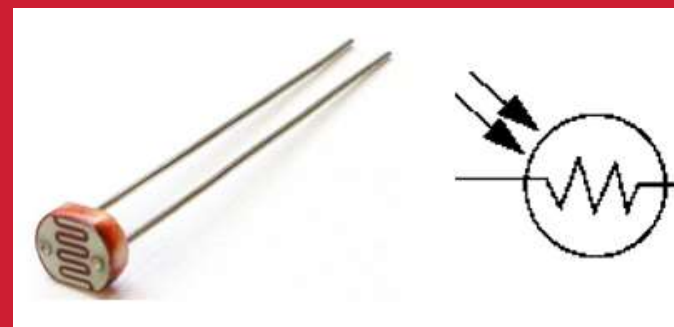
34 void loop(){
35     // 스위치 입력이 발생하였을 경우 실행
36     if(digitalRead(inputPin) == LOW){
37         // 현재의 시간을 startTime 변수에 넣는다.
38         startTime=millis();
39         // 스위치가 입력되는 동안 지연시킨다.
40         while(digitalRead(inputPin)==LOW);
41         // swCountTimer 변수에 스위치가 눌려진 시간을 넣는다.
42         // 여기까지 측정된 시간에서 앞서 저장한 시간이
43         // 스위치가 눌려진 시간이 된다
44         swCountTimer = millis() - startTime;
45         // LCD로 경과 시간을 출력한다..
46         delay(100);
47         // 모두 삭제
48         lcd.clear();
49         // 커서를 좌측 상단으로
50         lcd.setCursor(0,0);
51         lcd.print("ARnn time span");
52         // 커서를 두 번째 줄로
53         lcd.setCursor(0,1);
54         lcd.print(swCountTimer);
55         lcd.print(" ms");
56     }
57 }

```

161mss ← Bug!



# 6. Analog input





## 6. 아날로그 신호 입력

6.1     포텐쇼미터 입력 (가변저항기)

6.2     빛 입력 (CdS, LDR)

6.3     온도 측정 (LM35, TMP36)

6.4     수위 측정

6.5     아날로그 조이스틱

6.6     소리 입력

[DIY] TMP36 + 4-FND



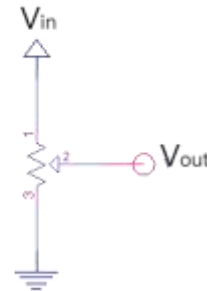
# 6.1 potentiometer

## 가변저항기



# 6.1 포텐쇼미터 (가변저항 조절)

## 포텐쇼미터 (Potentiometer)



- ✓ 회전, 직선 변위를 감지하는 센서
- ✓ 위치에 따라 저항 값이 변화함. (가변저항기)
- ✓ ADC를 이용하여 변화된 저항에 전압을 인가하여 전압의 변화를 감지

# 6.1.1 포텐쇼미터 (가변저항 조절)

## EX 6.1

### 포텐쇼미터 입력 (1/3)

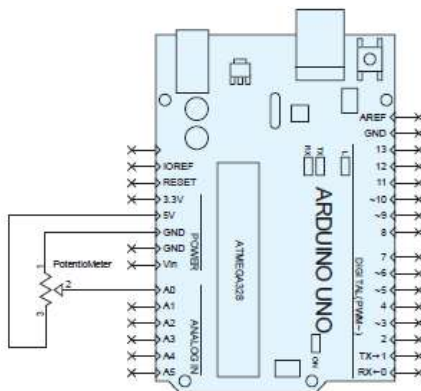
- 실습목표**
1. 포텐쇼미터를 회전에 따라 LED의 점멸 주기를 조절해 보자.
  2. 포텐쇼미터의 값을 아날로그 핀을 통하여 0~1023 범위로 읽는다.
  3. 이를 0~100의 범위의 숫자로 변경한다. (또는 0~10 kΩ, 0~5V)
  4. 변경된 숫자를 참고하여 LED의 듀티비를 조절한다.
  5. 현재 포텐쇼미터의 값을 시리얼 통신으로 출력한다.
- Hardware**
1. 실험에 사용할 포텐쇼미터는 10kΩ 사양의 3개의 핀이 있다. 1, 3번핀에서 측정되는 저항값을 포텐쇼미터의 회전에 따라 2번핀으로 나누게 된다. 즉 1, 3번핀 사이의 저항값을 포텐쇼미터가 회전을 하면 2번핀이 1, 2번핀 사이의 저항과 2, 3번핀 사이의 저항으로 나누게 된다.
  2. 이 때 1, 3번핀 양단에 전압차가 발생하면 옴의법칙에 의해서 전압은 저항값의 비율로 나뉘지게 된다. 즉 1, 2번핀의 저항값과 2, 3번핀의 저항값의 비가 5:5라면 전압도 5:5로 나뉜다.
  3. 포텐쇼미터 1번핀에 GND, 3번핀에 5V를 연결한다.
  4. 포텐쇼미터의 2번핀을 아날로그입력 0번핀(A0)에 연결한다.
  5. 포텐쇼미터를 회전시키면 1, 2번핀 사이의 저항이 변화한다. 저항의 변화에 따라 전압도 변화한다. 이 때 전압의 범위는 0~5V이다. 이를 ADC로 읽어 포텐쇼미터의 회전 각도를 알아낼 수 있다.

# 6.1.2 포텐쇼미터 (가변저항 조절)

EX 6.1

## 포텐쇼미터 입력 (2/3)

Hardware



Commands

- `analogRead`(아날로그 핀번호)

아날로그핀에서 아날로그 값을 읽는다. 0~5V사이의 전압을 0~1023사이의 값으로 표현한다.

- `map` (변수명, 범위1 최소값, 범위1 최대값, 범위2 최소값, 범위2 최대값)

변수명의 변수의 범위1의 범위와 범위2의 범위에 매칭시킨다. 즉 변수가 0~1023의 범위를 갖고 이를 0~100의 범위로 매칭하려면 '`map(변수명, 0, 1023, 0, 100)`'의 명령어로 매칭시킬 수 있다.

## 6.1.3 포텐쇼미터 (가변저항 조절)

### EX 6.1

### 포텐쇼미터 입력 (3/3)

- Sketch 구성
1. A0핀에 아날로그 입력을 받고 내장 LED인 13번 핀을 출력으로 사용한다.
  2. '`analogRead()`' 명령어로 포텐쇼미터 값을 읽는다.
  3. '`map()`' 명령어로 포텐쇼미터값(0~1023)과 LED의 듀티비(0~100)를 매칭시킨다.
  4. ADC 값과 듀티비를 시리얼통신으로 PC에 전송한다.
- 실행 결과
1. 포텐쇼미터를 회전시킬 때 마다 LED의 점등 주기가 변경된다.
  2. 시리얼 모니터에 'ADC Value: XXX, Duty cycle: XXX%'이 표시된다.

```

6 // 0번 아날로그핀을 포텐쇼미터 입력으로 설정한다.
7 const int potentiometerPin = 0;
8
9 //13번 핀에 연결되어 있는 내장 LED를 출력으로 사용한다.
10 const int ledPin = 13;
11
12 void setup() {
13 // 13번 핀을 출력으로 설정한다.
14 pinMode(ledPin, OUTPUT);
15 // 시리얼 통신을 설정한다.
16 Serial.begin(9600);
17 }

```

COM6

```

ADC Value: 157. Duty cycle: 15%
ADC Value: 158. Duty cycle: 15%
ADC Value: 155. Duty cycle: 15%
ADC Value: 155. Duty cycle: 15%
ADC Value: 155. Duty cycle: 15%
ADC Value: 155. Duty cycle: 15%

```

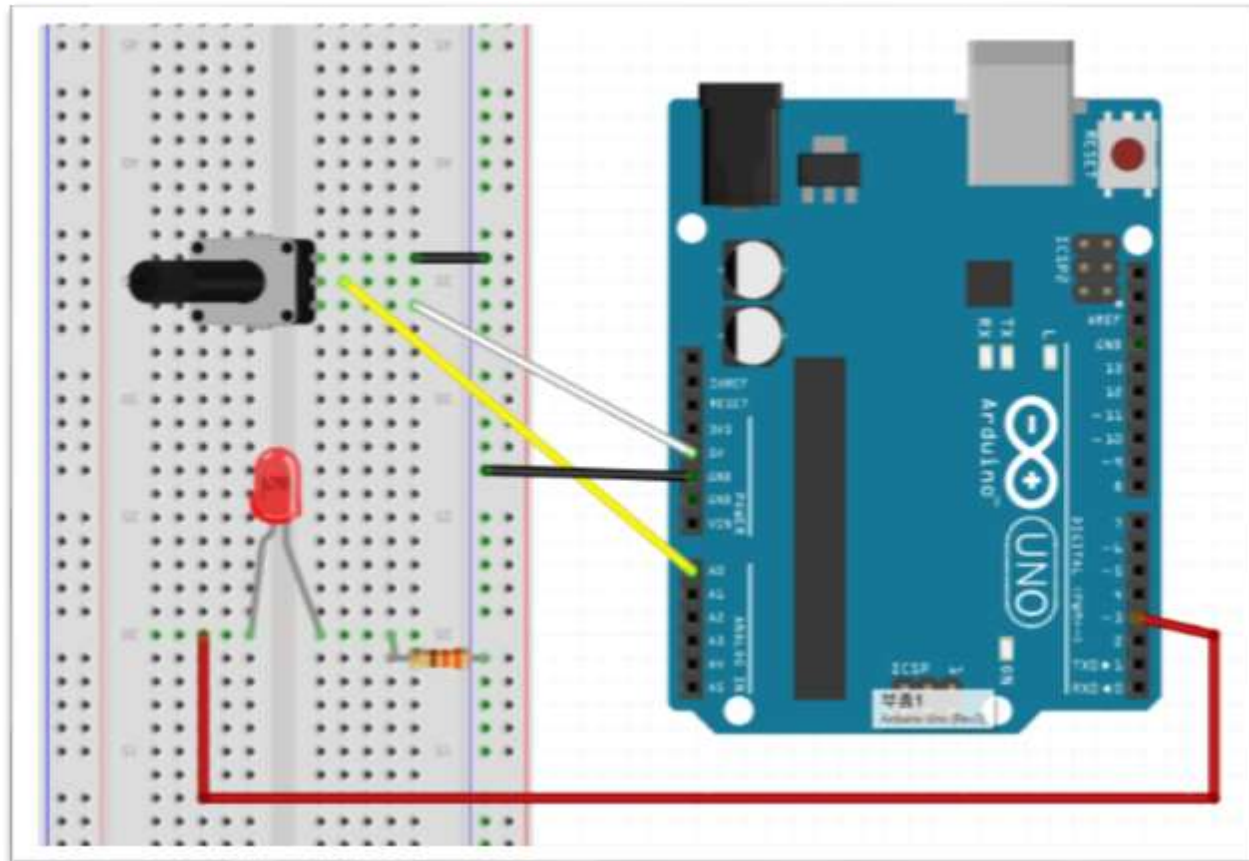
```

19 void loop(){
20
21     int adcValue; // 실제 센서로부터 읽은 값 (0~1023)
22     int duty;      // LED 점멸 주기 (0~100%)
23
24     // 포텐쇼미터 값을 읽는다.
25     adcValue = analogRead(potentiometerPin);
26     // 포텐쇼미터 값을 0~100의 범위로 변경한다.
27     duty = map(adcValue, 0, 1023, 0, 100);
28
29     // LED를 duty ms 만큼 점등한다.
30     digitalWrite(ledPin, HIGH);
31     delay(duty);
32
33     // 나머지 시간에는 소등시킨다.
34     digitalWrite(ledPin, LOW);
35     delay(100-duty);
36
37     // 시리얼 통신으로 ADC 값과 Duty를 출력한다.
38     Serial.print("ADC Value: ");
39     Serial.print(adcValue);
40     Serial.print(". Duty cycle: ");
41     Serial.print(duty);
42     Serial.println("");
43 }

```

# 6.1.5 포텐쇼미터 (가변저항 조절) - DIY

- DIY 응용 문제
1. 4.2절의 예제를 참고하여 PWM 단자를 이용하여 LED의 밝기를 조절해 보자.
- PWM을 지원하는 디지털 3 번 핀에 단색 LED를 연결. (220 ohm 저항 연결)





# 6.1.6 포텐쇼미터 (DIY: code-2, pwm)

```

6 // 0번 아날로그핀을 포텐쇼미터 입력으로 설정한다.
7 const int potentiometerPin = 0;
8
9 //13번 핀에 연결되어 있는 내장 LED를 출력으로 사용한다.
10 const int ledPin = 13;
11
12 // #3 pin is defined to PWM output pin
13 const int pwmOutputPin = 3;
14
15 void setup() {
16 // 13번 핀을 출력으로 설정한다.
17 pinMode(ledPin, OUTPUT);
18 // 시리얼 통신을 설정한다.
19 Serial.begin(9600);
20 }

```

COM5

```

ADC Value: 1023, Duty cycle: 100%, pwm: 255
ADC Value: 1023, Duty cycle: 100%, pwm: 255
ADC Value: 1023, Duty cycle: 100%, pwm: 255
ADC Value: 1023, Duty cycle: 100%, pwm: 255
ADC Value: 1023, Duty cycle: 100%, pwm: 255
ADC Value: 1022, Duty cycle: 99%, pwm: 254
ADC Value: 1023, Duty cycle: 100%, pwm: 255

```

```

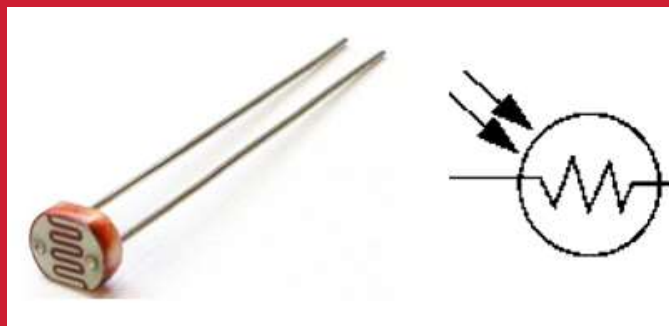
22 void loop(){
23   int adcValue; // 실제 센서로부터 읽은 값 (0~1023)
24   int duty;      // LED 점멸 주기 (0~100%)
25   int pwm;       // pwm 출력용
26
27   // 포텐쇼미터 값을 읽는다.
28   adcValue = analogRead(potentiometerPin);
29   // 포텐쇼미터 값을 0~100의 범위로 변경한다.
30   duty = map(adcValue, 0, 1023, 0, 100);
31
32   // LED를 duty ms 만큼 점등한다.
33   digitalWrite(ledPin, HIGH);
34   delay(duty);
35   // 나머지 시간에는 소등시킨다.
36   digitalWrite(ledPin, LOW);
37   delay(100-duty);
38
39   // pwmOutputPin Led ON
40   pwm = map(adcValue, 0, 1023, 0, 255);
41   analogWrite(pwmOutputPin,pwm);
42   // 시리얼 통신으로 ADC 값과 Duty를 출력한다.
43   Serial.print("ADC Value: ");
44   Serial.print(adcValue);
45   Serial.print(". Duty cycle: ");
46   Serial.print(duty);
47   Serial.println("%");
48 }

```



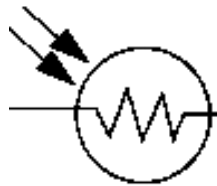
## 6.2 CdS, LDR

# 조도센서



# 6.2 조도 센서 (빛의 밝기 측정)

## CdS 센서



- ✓ CdS 분말을 세라믹 기판 위에 압축하여 제작
- ✓ 빛이 강할 수록 저항 값이 감소 → 광 가변저항
- ✓ ADC를 이용하여 변화된 저항에 전압을 인가하여 전압의 변화를 감지
- ✓ 자동 조명장치, 조도 측정 등에 사용

## 럭스

다른 뜻에 대해서는 [Lux](#) 문서를 참조하십시오.

럭스(lux, 기호 **lx**)는 빛의 **조명도**를 나타내는 SI 단위이다. 럭스는 루멘에서 유도

$$1 \text{ lx} = 1 \text{ lm/m}^2 = 1 \text{ cd}\cdot\text{sr}\cdot\text{m}^{-2}$$

럭스의 예 [\[편집\]](#)

I 밝기차	예
10 <sup>-5</sup> lux	가장 밝은 별(시리우스)의 빛 <sup>[1]</sup>
10 <sup>-4</sup> lux	하늘을 덮은 완전한 별빛 <sup>[1]</sup>
0.002 lux	대기광이 있는 달 없는 맑은 밤 하늘 <sup>[1]</sup>
0.01 lux	초승달
0.27 lux	맑은 밤의 보름달 <sup>[1][2]</sup>
1 lux	열대 위도를 덮은 보름달 <sup>[3]</sup>
3.4 lux	맑은 하늘 아래의 어두운 황혼 <sup>[4]</sup>
50 lux	거실 <sup>[5]</sup>
80 lux	복도/화장실 <sup>[6]</sup>
100 lux	매우 어두운 낮 <sup>[1]</sup>
320 lux	권장 오피스 조명 (오스트레일리아) <sup>[7]</sup>
400 lux	맑은 날의 해돋이 또는 해넘이
1000 lux	인공 조명 <sup>[1]</sup> ; 일반적인 TV 스튜디오 조명
10,000–25,000 lux	낮 (직사광선이 없을 때) <sup>[1]</sup>
32,000–130,000 lux	직사광선

# 6.2.1 조도 센서 (빛의 밝기 측정)

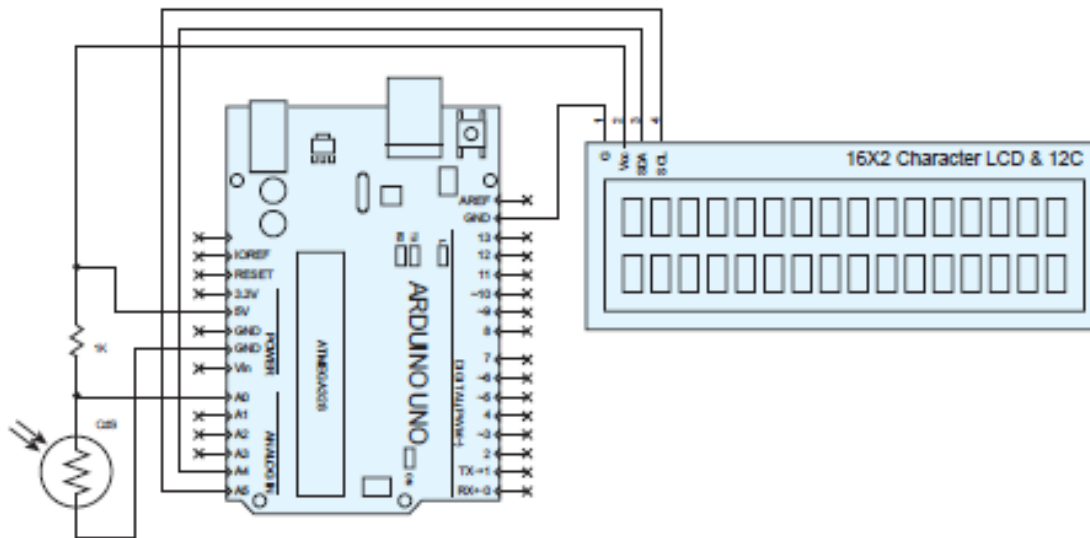
## EX 6.2

## 빛 입력 (1/3)

**실습목표** CdS 셀을 이용하여 조도를 측정해 보자.

1. CdS 셀로 측정된 조도를 아날로그 핀을 통하여 0~1023 범위로 읽는다.
2. ADC 값을 LCD 모듈로 0~100%의 범위로 출력한다.

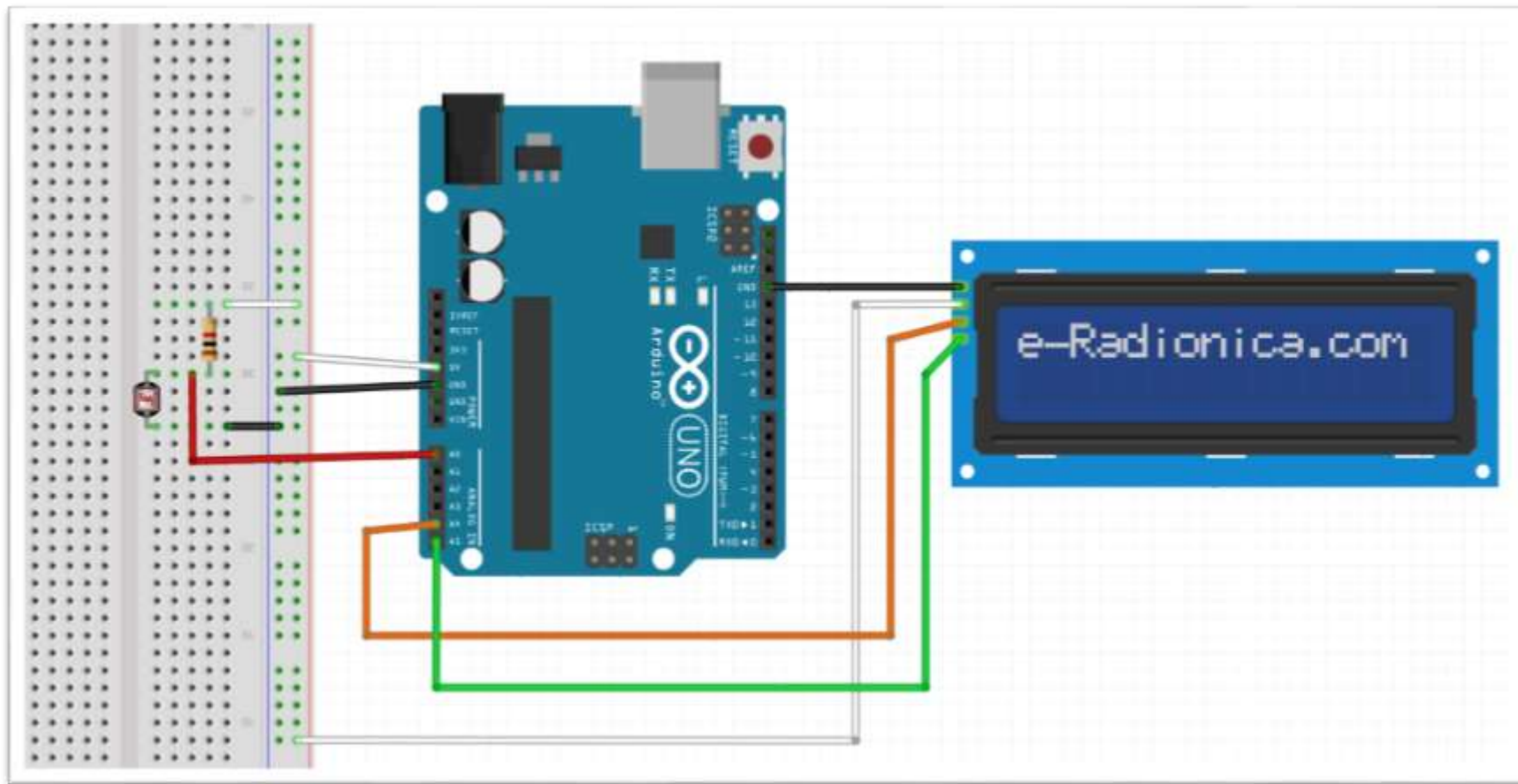
- Hardware**
1. CdS셀과 1k $\Omega$ 저항을 연결한 뒤 저항의 한쪽 끝은 5V에 CdS셀의 한쪽 끝은 GND에 연결한다.
  2. 저항과 CdS셀 사이를 아날로그입력핀 A0에 연결한다.
  3. I2C LCD 모듈의 Vcc, GND를 Arduino의 5V, GND에 연결한다.
  4. I2C LCD 모듈의 SDA는 A4에 SCL은 A5에 연결한다.



# 6.2.1 조도 센서 (빛의 밝기 측정)

EX 6.2

빛 입력 (1/3)

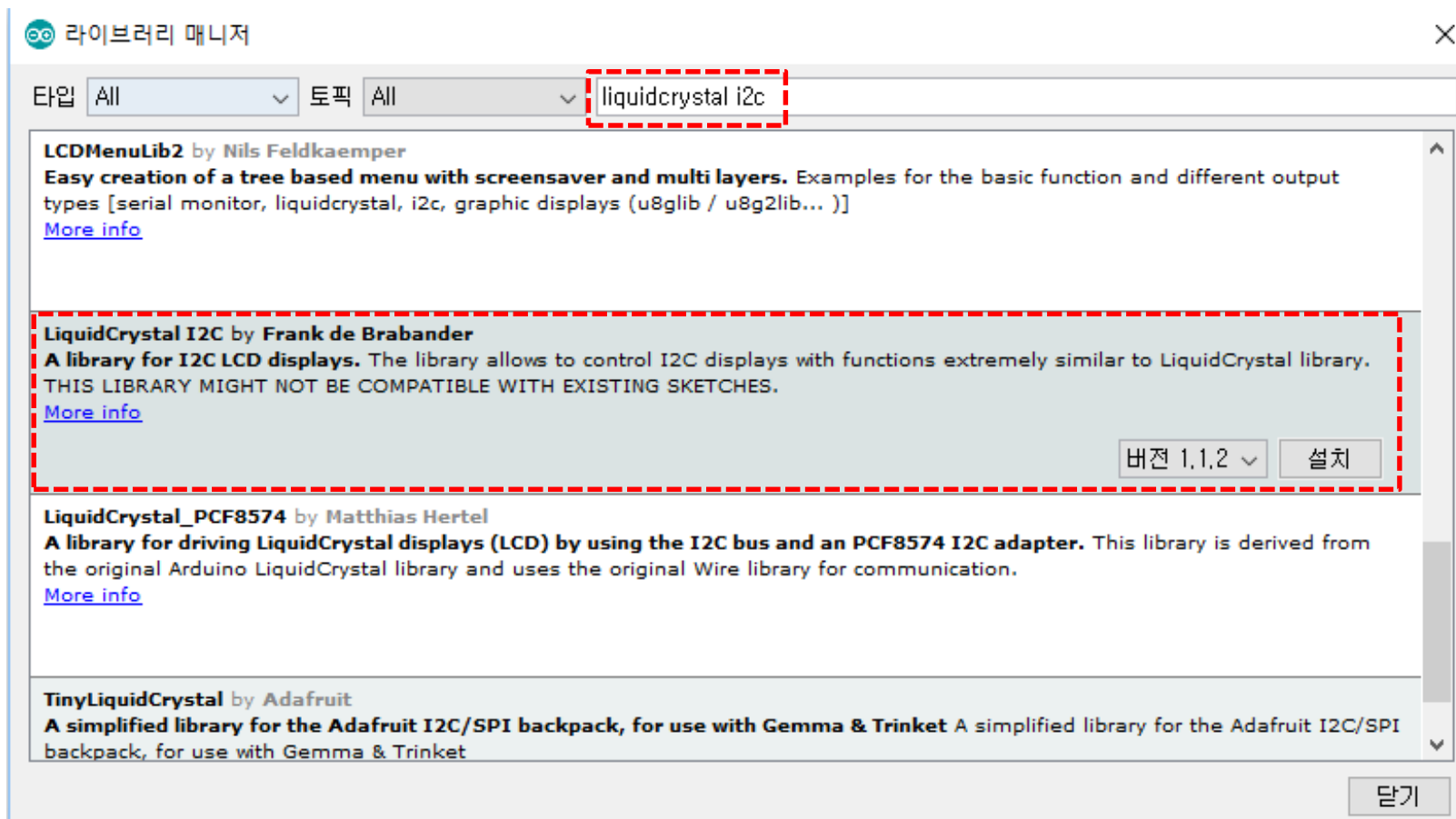


**CdS는 10 kΩ 저항과 직렬로 연결**

## 3.2.3 I<sup>2</sup>C를 이용한 LCD 출력

라이브러리 매니저를 이용하여 I<sup>2</sup>C LCD용 라이브러리(LiquidCrystal I2C)를 설치

스케치 > 라이브러리 포함하기 > 라이브러리 관리



## 6.2.2 조도 센서 (빛의 밝기 측정)

### EX 6.2

### 빛 입력 (2/3)

#### Commands

- **analogRead(아날로그 핀번호)**

아날로그 핀에서 아날로그 값을 읽는다. 0~5V사이의 전압을 0~1023 사이의 값으로 표현한다.

- **map(변수명, 범위1 최소값, 범위1 최대값, 범위2 최소값, 범위2 최대값)**

변수명의 변수의 범위1의 범위와 범위2의 범위에 매칭시킨다. 즉 변수가 0~100의 범위를 갖고

이를 50~200의 범위로 매칭하려면 'map(변수명, 0, 100, 50, 200)'의 명령어로 매칭시킬 수 있다.

- **LiquidCrystal\_I2C(I2C 주소, 가로 글자수, 세로 글자수)**

→ LCD 모듈이 연결된 I2C 주소와 LCD의 가로, 세로 글자수를 설정한다.

- **lcd.init( );** LCD 모듈을 설정한다.

- **lcd.clear( );** lcd란 이름의 LCD 모듈의 화면의 모든 표시를 지우고 커서를 왼쪽 위로 옮긴다.

- **lcd.home( );** lcd란 이름의 LCD 모듈의 커서를 왼쪽 위로 옮긴다.

- **lcd.setCursor(행, 열);** lcd란 이름의 LCD 모듈의 커서를 원하는 위치로 이동시킨다.

- **lcd.print(데이터);** lcd란 이름의 LCD 모듈에 데이터를 출력한다.

- **lcd.noBacklight();** lcd란 이름의 LCD 모듈의 백라이트를 소등한다.

- **lcd.backlight();** lcd란 이름의 LCD 모듈의 백라이트를 점등한다.

## 6.2.3 조도 센서 (빛의 밝기 측정)

EX 6.2

### 빛 입력 (3/3)

Sketch 구성

1. CdS 센서로부터 읽은 빛의 밝기를 I2C 16X2 LCD 모듈로 출력하기 위해 CdS 센서 입력 핀 설정과 LCD 모듈 설정을 한다.
2. CdS 센서로부터 읽은 ADC값을 LCD에 출력하고 밝기를 %로 나타낸다.

실습 결과 ADC 값과 조도값이 표시된다..

```
ADC: 500
Illuminance: 50 %
```





## 6.2.4 조도 센서 (빛의 밝기 측정) : code-1

```

6 // I2C 통신 라이브러리 설정
7 #include <Wire.h>
8 // I2C LCD 라이브러리 설정
9 #include <LiquidCrystal_I2C.h>
10
11 // LCD I2C address 설정
12 LiquidCrystal_I2C lcd(0x3f, 16, 2);
13
14 // 0번 아날로그핀을 CdS 셀 입력으로 설정한다.
15 const int CdSPin = 0;
16

```

```

17 void setup() {
18
19 // 16X2 LCD 모듈 설정하고 백라이트를 켜다.
20 lcd.init(); // LCD 설정
21 lcd.backlight();
22
23 // 메시지를 표시한다.
24 lcd.print("ex 6.2");
25 lcd.setCursor(0, 1);
26 lcd.print("CdS Cell Test");
27 // 3초동안 메시지를 표시한다.
28 delay(3000);
29
30 // 모든 메시지를 삭제한 뒤
31 // 숫자를 제외한 부분들을 미리 출력시킨다.
32 lcd.clear();
33 lcd.setCursor(0, 0);
34 lcd.print("ADC : ");
35 lcd.setCursor(0, 1);
36 lcd.print("Illuminance:");
37 lcd.setCursor(15, 1);
38 lcd.print("%");
39 }

```

## 6.2.4 조도 센서 (빛의 밝기 측정) : code-2

```

41 void loop(){
42
43   int adcValue; // 실제 센서로부터 읽은 값 (0~1023)
44   int illuminance; // 현재의 밝기. 0~100%
45
46   // CdS cell을 통하여 입력되는 전압을 읽는다.
47   adcValue = analogRead(CdSPin);
48   // 아날로그 입력 값을 0~100의 범위로 변경한다.
49   illuminance = map(adcValue, 0, 1023, 100, 0);
50
51   // 전에 표시했던 내용을 지우고
52   // LCD에 ADC 값과 밝기를 출력한다.
53   // 지우지 않으면 이전에 표시했던 값이 남게 된다.
54

```



**ARnn\_cds.png** 로 저장...

```

55   // 전에 표시했던 내용을 지운다.
56   lcd.setCursor(9,0);
57   lcd.print(" ");
58   // ADC 값을 표시한다
59   lcd.setCursor(9,0);
60   lcd.print(adcValue);
61
62   // 전에 표시했던 내용을 지운다.
63   lcd.setCursor(13,1);
64   lcd.print(" ");
65   // 밝기를 표시한다
66   lcd.setCursor(12,1);
67   lcd.print(illuminance);
68
69   delay(1000);
70 }

```

## DIY

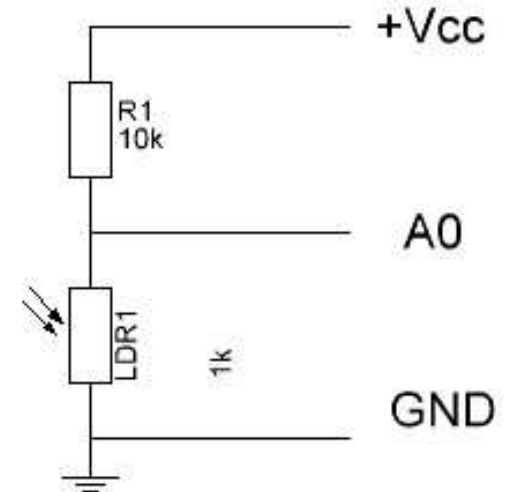
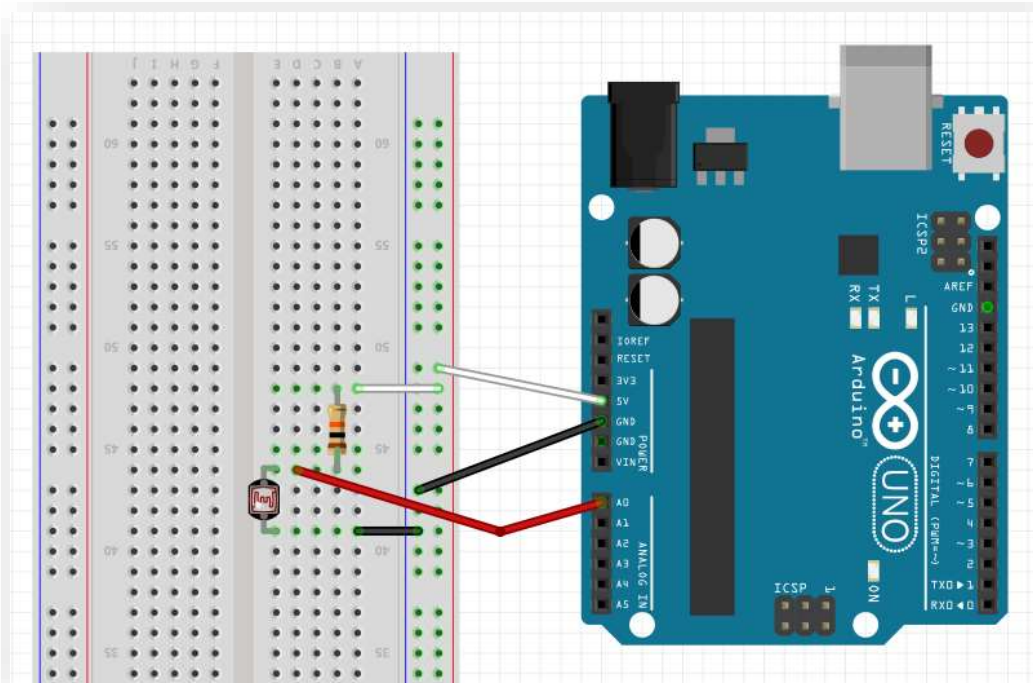
## 응용 문제

1. 손으로 가렸을 때 LCD 모듈의 백라이트가 켜지고, 가리지 않았을 때 백라이트가 꺼지도록 수정하여 보자.
2. 손으로 가렸을 때 13번핀에 연결된 LED가 켜지고, 가리지 않았을 때 꺼지도록 수정하여 보자.

위의 1, 2가 동시에 실행되는 아두이노 코드를 완성하시오.

→ ARnn\_cds\_project.ino 로 저장 제출

## CdS 센서 회로



**Parts : photocell CdS, R (10 kΩ X 1)**

저항과 광센서 사이에서 전압 값을 **A0**로 측정

# [참고] 조도 센서 (빛의 밝기 정밀 측정)

## CdS 센서 회로 - 측정 1.

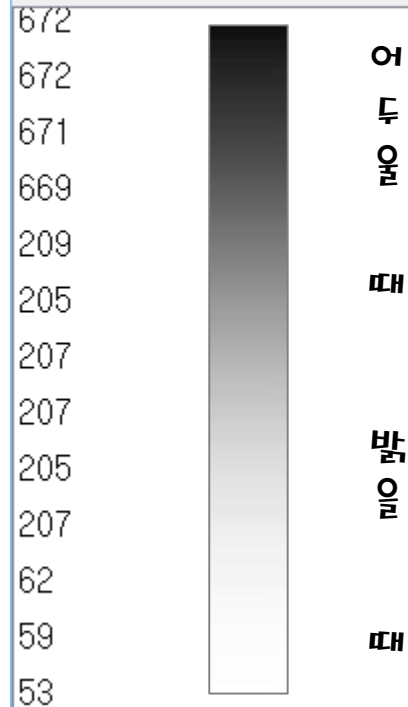
CdS\_start

```

1 #define CDS_INPUT 0
2
3 void setup() {
4   Serial.begin(9600);
5 }
6
7 void loop() {
8
9   int value = analogRead(CDS_INPUT);
10  Serial.println(value);
11
12  delay(1000);
13 }
14

```

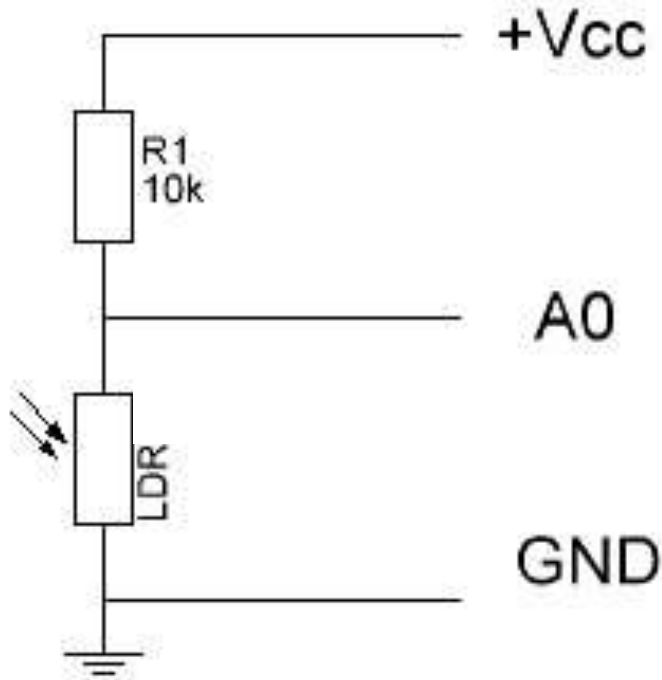
COM11 (Arduino/Genuino Uno)



어두우면 측정 값이 커지고 밝을수록 값이 작아진다 ???

# [참고] 조도 센서 (빛의 밝기 정밀 측정)

## CdS 센서 회로 분석 (1/2)



**LDR's (Light dependent resistors) have a low resistance in bright light and a high resistance in the darkness.**

**If you would use the LDR as the lower part of a voltage divider, then in darkness there would be a high voltage over the LDR, while in bright light, there would be a low voltage over that resistor.**

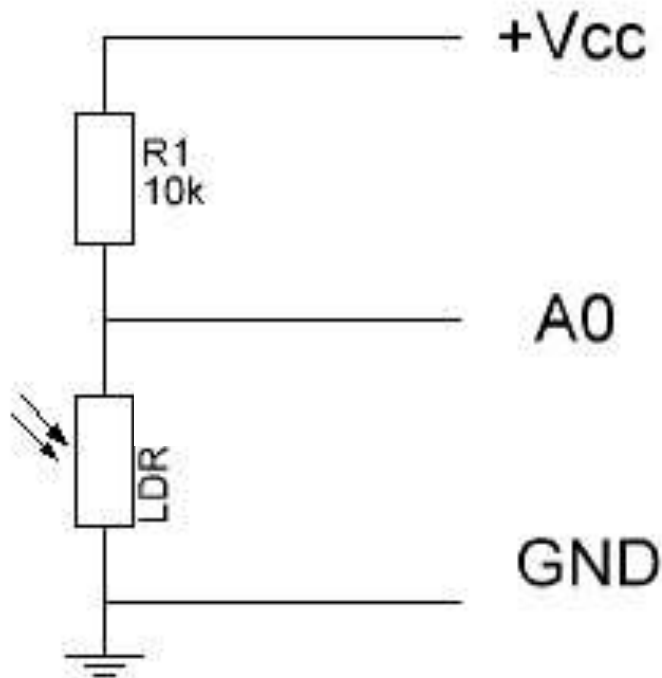
어두우면 측정 값이 작아지고 밝을수록 값이 커져야 된다.  
그리고 측정 값은 **lux**로 표현된다.

$$V_{out} = \frac{R_{ldr}}{R_1 + R_{ldr}} * V_{cc}$$

**A0에서 측정되는 LDR**  
**양단의 전압 = V<sub>out</sub>**

# [참고] 조도 센서 (빛의 밝기 정밀 측정)

## CdS 센서 회로 분석 (2/2)



$$(a) \quad V_{out} = \frac{R_{ldr}}{(R_1 + R_{ldr})} * V_{CC} ,$$

$$(b) \quad R_{ldr} = \frac{10 * V_{out}}{(5 - V_{out})} (k\Omega) ,$$

$$(c) \quad V_{out} = value * V_{CC}/1023 ,$$

$$(d) \quad Lux = \frac{500}{R_{ldr}} ,$$

$$(e) \quad Lux = \left( \frac{2500}{V_{out}} - 500 \right) / 10 (lux) .$$

$$V_{out} = \frac{R_{ldr}}{R_1 + R_{ldr}} * V_{cc}$$

**A0**에서 측정되는 **LDR**  
양단의 전압 = **V<sub>out</sub>**

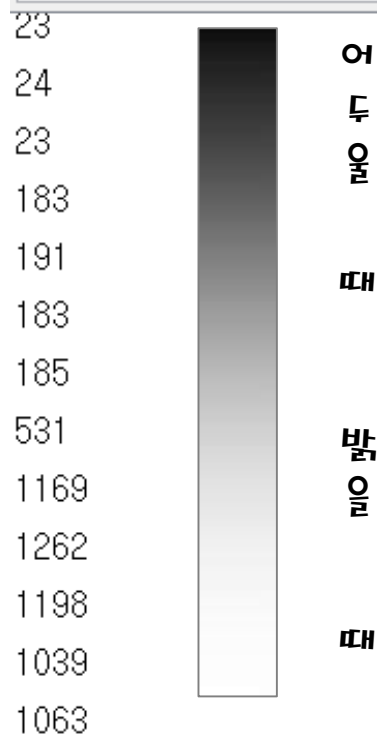
# [참고] 조도 센서 (빛의 밝기 정밀 측정)

## CdS 센서 회로 - 측정 2.

```

sketch08_CdS2
1 // lux
2 #define CDS_INPUT 0
3
4 void setup() {
5   Serial.begin(9600);
6 }
7 void loop() {
8   int value = analogRead(CDS_INPUT);
9   Serial.println(int(luminosity(value)));
10  delay(1000);
11 }
12
13 //Voltage to Lux
14 double luminosity (int RawADC0){
15   double Yout=RawADC0*5.0/1023; // 5/1023 (Vin = 5 V)
16   double lux=(2500/Yout-500)/10;
17   // lux = 500 / Rldr, Yout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
18   return lux;
19 }
  
```

COM11 (Arduino/Genuino Uno)



밝을수록 측정 값이 커지고  
어두울수록 값이 작아진다 !!!





# 온도센서



## 6.3 온도 센서 (주변 온도 측정)

LM35



- ✓ 온도 측정을 위한 센서 ( $-55^{\circ}\text{C}$  and  $150^{\circ}\text{C}$ )
- ✓ 전원과 접지를 연결하면 Vout에 0~500도까지 0.01V 단위로 전압 출력이 발생
- ✓ ADC를 이용하여 이 값을 읽어 온도를 측정
- ✓ [단점] 온도 측정 오차가 크다. (TMP36이 정확)

# 6.3 온도 센서 (주변 온도 측정)

Google

lm35 voltage temperature relation



전체

이미지

동영상

뉴스

더보기

설정

도구

검색결과 약 50,700개 (0.40초)



**LM35.** The **LM35 temperature** sensor provides an output of 10mV per degree Celsius, with an accuracy of 0.5°C at 25°C. It can be powered by any DC **voltage** in the range 4 – 30v. The operating range is -55°C to +150°C. The output of the LM 35 is 10 mV (0.01 volts) per degree Celsius.



LM35 Temperature detector

[LM35 Precision temperature measurement - Back](#)

[www.magics-notebook.com/lm35.html](http://www.magics-notebook.com/lm35.html)

 이 결과에 관한 정보  사용자 의견

[LM35 temperature equation? - Arduino Forum](#)

<https://forum.arduino.cc/index.php?topic=99421.0>  이 페이지 번역하기

2012. 4. 1. - 답글 4개 - 작성자 3명

**Equation** to get **temperature** using **LM35**: temp = (5.0 ... supply you use, the analog **voltage** reading will range from about 0V (ground) to about ...

<b>LM35: Temperature</b> Readings are not right	게시물 15개	2015년 4월 5일
Guide: accurately read an <b>LM35</b>	게시물 15개	2010년 2월 7일
<b>LM35</b> thermometer	게시물 15개	2008년 10월 18일

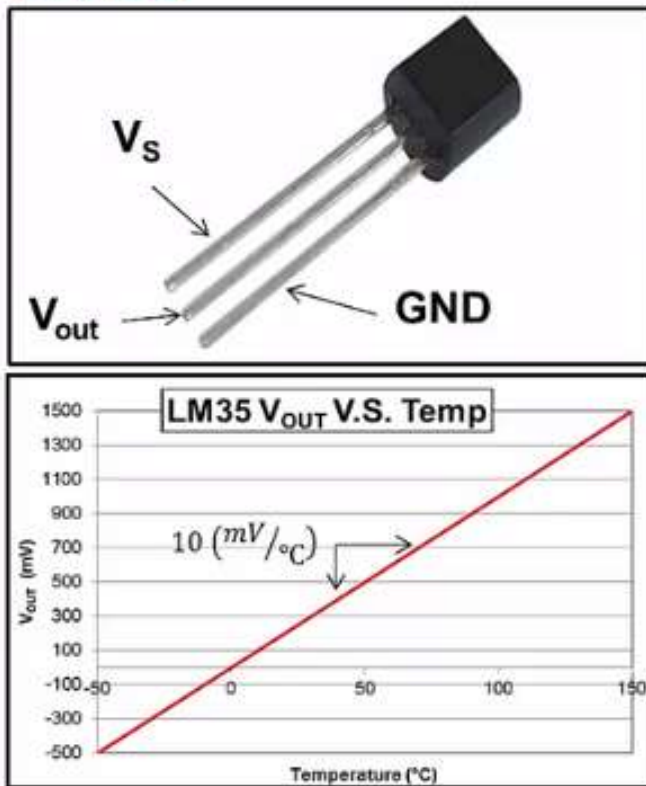
[forum.arduino.cc](#) 검색결과 더보기

[what is conversion system of LM35 \(temperature sensor\) in Celsius ...](#)

## 6.3 온도 센서 (주변 온도 측정)

### LM35 온도-전압 특성

#### LM35



- Three-Pin
  - TO-92 Package
  - Easy to Use
  - 4V-20V Operating Range
  - 60 $\mu$ A Max Current Draw
- Analog Output
  - 0.5 $^{\circ}C$  Accuracy at 25 $^{\circ}C$
  - Easily read by Arduino
  - Highly Linear Transfer Function
  - 10 ( $mV/^{\circ}C$ ) Slope

✓ 전원과 접지를 연결하면  $V_{out}$ 에 0~500  $^{\circ}C$  까지 0.01V 단위로 전압 출력(0~5000mV)이 발생

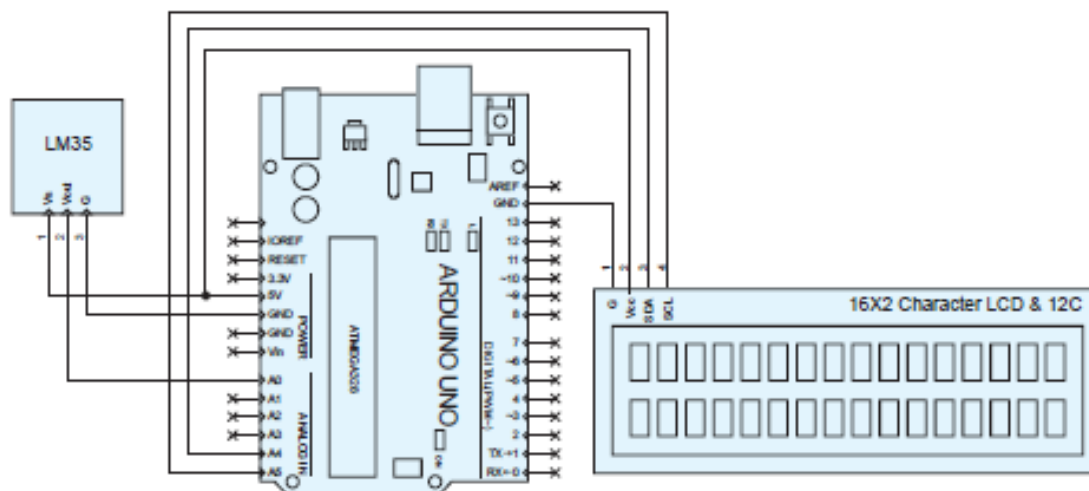
# 6.3.1 온도 센서 (주변 온도 측정)

EX 6.3

온도 측정 (1/3)

- 실습목표**
1. LM35 센서로부터 현재 온도를 아날로그 입력핀으로 측정한다.
  2. 측정된 값을 LCD에 표시해 보자.

- Hardware**
1. LM35의 Vs와 G 핀을 Arduino의 5V와 GND에 연결한다.
  2. LM35의 Vout을 아날로그입력핀 A0에 연결한다.
  3. I2C LCD 모듈의 Vcc, GND를 Arduino의 5V, GND에 연결한다.
  4. I2C LCD 모듈의 SDA는 A4에 SCL은 A5에 연결한다.



## 6.3.2 온도 센서 (주변 온도 측정)

EX 6.3

### 온도 측정 (2/3)

#### Commands

- `analogRead`(아날로그 핀번호)

아날로그 핀에서 아날로그 값을 읽는다. 0~5V사이의 전압을 0~1023 사이의 값으로 표현한다.

- `map`(변수명, 범위1 최소값, 범위1 최대값, 범위2 최소값, 범위2 최대값)

변수명의 변수의 범위1의 범위와 범위2의 범위에 매칭시킨다. 즉 변수가 0~100의 범위를 갖고 이를 50~200의 범위로 매칭하려면 '`map(변수명, 0, 100, 50, 200)`'의 명령어로 매칭시킬 수 있다.

- `LiquidCrystal_I2C`(I2C 주소, 가로 글자수, 세로 글자수)

LCD 모듈이 연결된 I2C 주소와 LCD의 가로, 세로 글자수를 설정한다.

- `lcd.init( );` LCD 모듈을 설정한다.

- `lcd.clear( );` lcd란 이름의 LCD 모듈의 화면의 모든 표시를 지우고 커서를 왼쪽 위로 옮긴다.

- `lcd.home( );` lcd란 이름의 LCD 모듈의 커서를 왼쪽 위로 옮긴다.

- `lcd.setCursor(행, 열);` lcd란 이름의 LCD 모듈의 커서를 원하는 위치로 이동시킨다.

- `lcd.print(데이터);` lcd란 이름의 LCD 모듈에 데이터를 출력한다.

- `lcd.noBacklight();` lcd란 이름의 LCD 모듈의 백라이트를 소등한다.

- `lcd.backlight();` lcd란 이름의 LCD 모듈의 백라이트를 점등한다.

## 6.3.3 온도 센서 (주변 온도 측정)

EX 6.3

온도 측정 (3/3)

Sketch 구성

1. LM35 입력을 받기 위한 아날로그 핀을 설정한다.
2. ADC로 읽은 값과 실제 온도와의 관계는 연산을 통하여 계산한다.
3. I2C LCD 모듈에 ADC값과 현재 온도를 출력한다.
4. 온도를 출력 할 때 '°' 기호는 표 3.1 LCD 문자 코드표에서 찾아 코드를 이용하여 출력한다.

실행 결과

ADC 값과 온도가 표시된다.

```
ADC: 250
Temp. is 25 °C
```



```
ex_6_3
1 /*
2  예제 6.3
3  LM35를 이용한 온도 측정
4  */
5
6 // I2C 통신 라이브러리 설정
7 #include <Wire.h>
8 // I2C LCD 라이브러리 설정
9 #include <LiquidCrystal_I2C.h>
10
11 // LCD I2C address 설정
12 // PCF8574:0x27, PCF8574A:0x3F
13 LiquidCrystal_I2C lcd(0x27, 16, 2);
14
15 // 0번 아날로그핀을 LM35 입력으로 설정한다.
16 const int LM35Pin = 0;
17
```

```
18 void setup() {
19
20   // 16X2 LCD 모듈 설정하고 백라이트를 켜다.
21   lcd.init(); // LCD 설정
22   lcd.backlight();
23
24   // 메시지를 표시한다.
25   lcd.print("ex 6.3");
26   lcd.setCursor(0, 1);
27   lcd.print("Checking Temp.");
28
29   // 3초동안 메시지를 표시한다.
30   delay(3000);
31
32   // 모든 메시지를 삭제한 뒤
33   // 숫자를 제외한 부분들을 미리 출력시킨다.
34   lcd.clear();
35   lcd.setCursor(0, 0);
36   lcd.print("ADC : ");
37   lcd.setCursor(0, 1);
38   lcd.print("Temp. is ");
39
40   //LCD 문자표에서 '°' 기호를 직접 써준다
41   lcd.setCursor(13, 1);
42   lcd.write(B11011111); // '°' 기호 문자코드
43   lcd.setCursor(14, 1);
44   lcd.print("C"); // 'C'를 표시한다.
45 }
```



```

47 void loop(){
48   int adcValue; // 실제 센서로부터 읽은 값 (0~1023)
49   long temp; // 현재의 온도
50
51   // LM35의 Vout을 읽는다.
52   adcValue = analogRead(LM35Pin);
53   // 온도값으로 환산한다. 오버플로우 방지를 위하여 500L로 표시한다.
54   // 500L의 경우 500을 32비트 long 형태의 숫자로 나타내 준다.
55   temp = (adcValue * 500L) / 1023;
56
57   // 전에 표시했던 내용을 지우고 LCD에 ADC 값과 온도를 출력한다.
58   // 지우지 않으면 이전에 표시했던 값이 남게 된다.
59   lcd.setCursor(9,0);
60   lcd.print(" ");
61   // ADC 값을 표시한다
62   lcd.setCursor(9,0);
63   lcd.print(adcValue);
64
65   // 전에 표시했던 내용을 지운다.
66   lcd.setCursor(10,1);
67   lcd.print(" ");
68   // 온도를 표시한다
69   lcd.setCursor(10,1);
70   lcd.print(temp);
71
72   delay(2000);
73 }

```



## 6.3.5 온도 센서 (주변 온도 측정)

DIY

예제 6.2를 참고하여 LCD에 현재 온도, 조도를 함께 표시해 보자.

응용 문제

아두이노 코드를 완성하시오.

→ ARnn\_cds\_lm35.ino 로 저장하고 제출



→ ARnn\_cds\_lm35.png 로 저장하고 제출



# [Practice]

## ◆ [wk10]

- **Arduino : Analog input I.**
- **Complete your project**
- **Submit folder : Arnn\_Rpt09**

# wk10 : Practice-09 : ARnn\_Rpt09

## ◆ [Target of this week]

- Complete your works
- Save your outcomes
- Upload all in github.

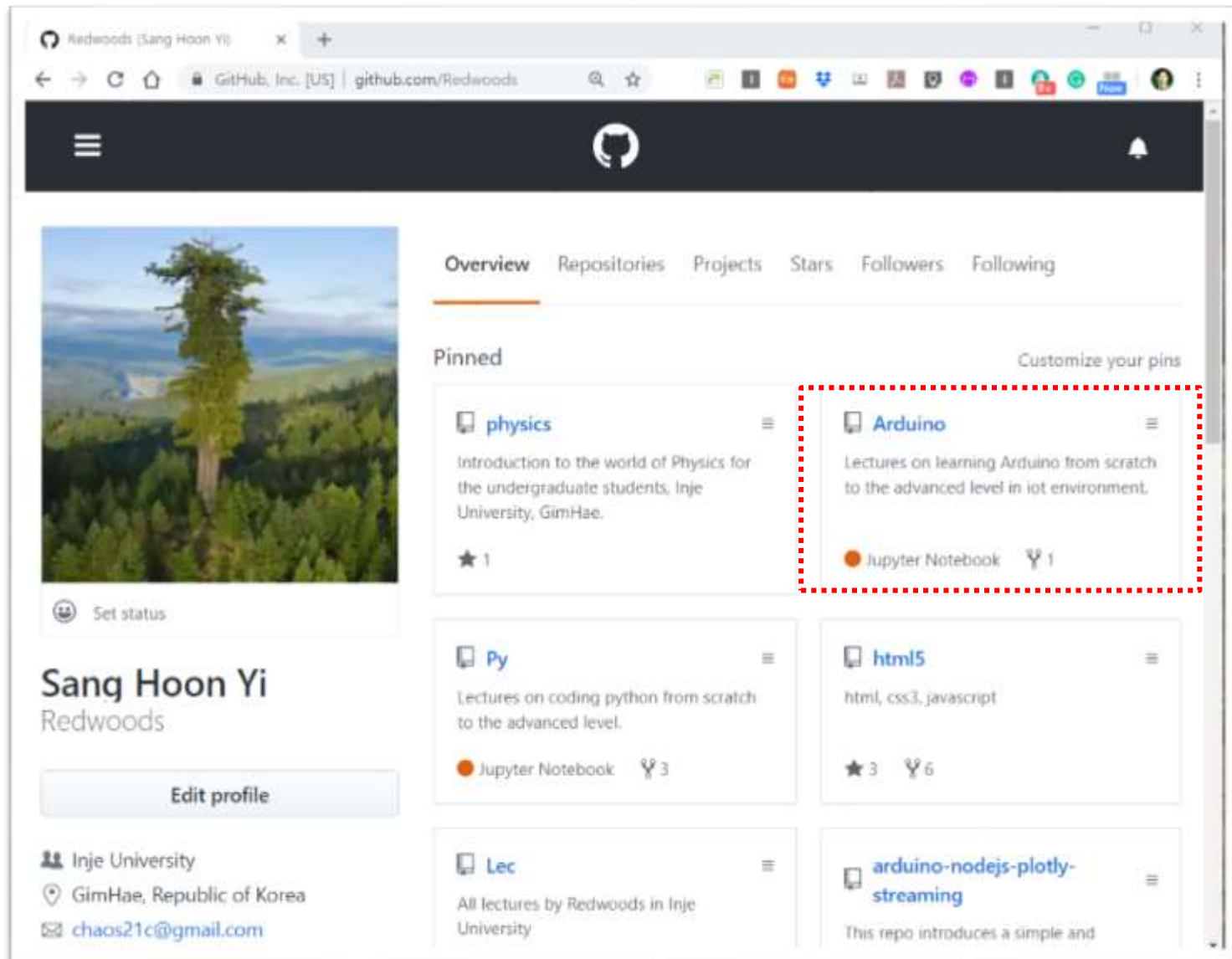
제출폴더명 : **ARnn\_Rpt09**

### 제출할 파일들

- ① **ARnn\_pwm.png**
- ② **ARnn\_pwm.ino**
- ③ **ARnn\_cds.png**
- ④ **ARnn\_cds\_project.ino**
- ⑤ **ARnn\_cds\_lm35.ino**
- ⑥ **ARnn\_cds\_lm35.png**
- ⑦ **\*.ino**

## ● References & good sites

- ✓ <http://www.nodejs.org/ko> Node.js
- ✓ <http://www.arduino.cc> Arduino Homepage
- ✓ <http://www.w3schools.com> By w3schools
- ✓ <http://www.github.com> GitHub
- ✓ <http://www.google.com> Googling



Redwoods (Sang Hoon Yi)

GitHub, Inc. [US] | github.com/Redwoods

Overview Repositories Projects Stars Followers Following

Pinned Customize your pins

**physics**  
Introduction to the world of Physics for the undergraduate students, Inje University, GimHae.  
★ 1

**Arduino**  
Lectures on learning Arduino from scratch to the advanced level in iot environment.  
Jupyter Notebook 🍴 1

**Py**  
Lectures on coding python from scratch to the advanced level.  
Jupyter Notebook 🍴 3

**html5**  
html, css3, javascript  
★ 3 🍴 6

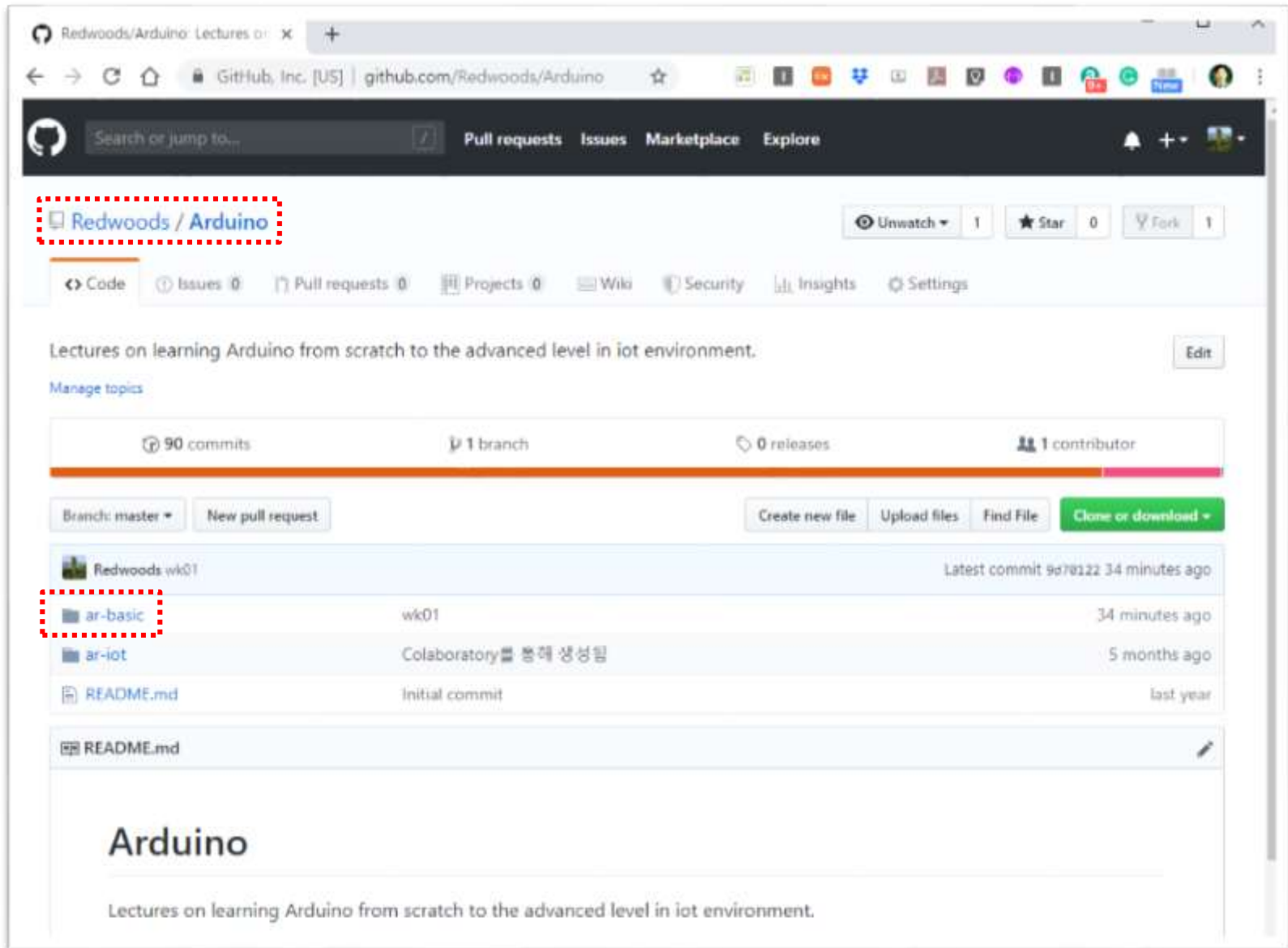
**Lec**  
All lectures by Redwoods in Inje University

**arduino-nodejs-plotly-streaming**  
This repo introduces a simple and

**Sang Hoon Yi**  
Redwoods

Edit profile

Inje University  
GimHae, Republic of Korea  
chaos21c@gmail.com



The screenshot shows the GitHub repository page for **Redwoods/Arduino**. The repository description is "Lectures on learning Arduino from scratch to the advanced level in iot environment." The repository has 90 commits, 1 branch, 0 releases, and 1 contributor. The file list includes **ar-basic** (34 minutes ago), **ar-iot** (5 months ago), and **README.md** (last year). The **ar-basic** file is highlighted with a red dashed box. The repository is currently on the **master** branch.

Redwoods / Arduino

Unwatch 1 Star 0 Fork 1

Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

Lectures on learning Arduino from scratch to the advanced level in iot environment. Edit

Manage topics

90 commits 1 branch 0 releases 1 contributor

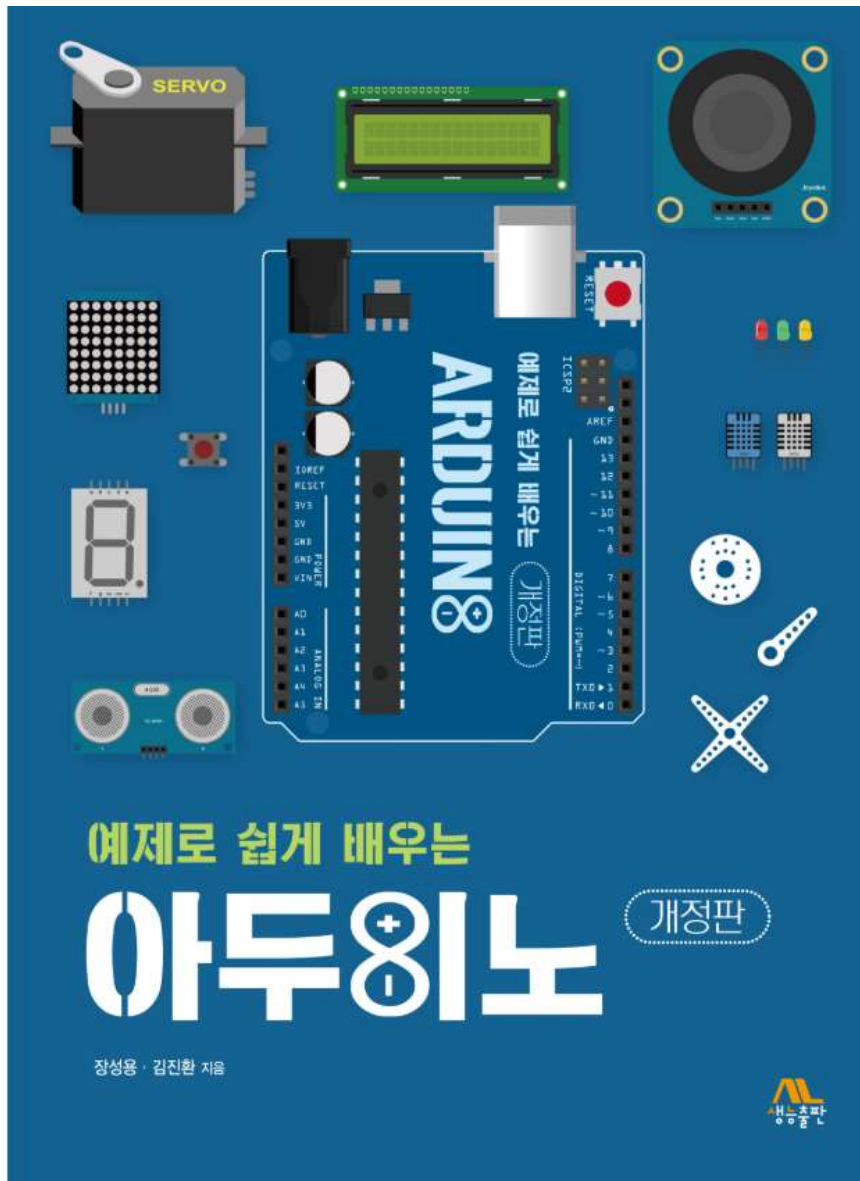
Branch: master New pull request Create new file Upload files Find File Clone or download

File	Commit	Time
Redwoods (wk01)	Latest commit 9d70122	34 minutes ago
ar-basic	wk01	34 minutes ago
ar-iot	Colaboratory를 통해 생성됨	5 months ago
README.md	Initial commit	last year

README.md

## Arduino

Lectures on learning Arduino from scratch to the advanced level in iot environment.

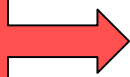




# 아두이노 키트(Kit)



web



<https://www.devicemart.co.kr/goods/view?no=12170416>

# 아두이노 키트(Kit) : Part-1

## 아두이노 레벨업키트(골드) 구성품

						
아두이노 UNO	USB 케이블	830핀 브레드보드	미니 브레드보드	점퍼와이어 세트	듀폰케이블 M/F	듀폰케이블 M/M
						
저항 220Ω	저항 1KΩ	저항 10KΩ	가변저항 10KΩ	빨강 LED	녹색 LED	파랑 LED
						
노랑 LED	RGB LED (CA)	RGB LED 모듈	1digit FND (CA)	4digit FND (CA)	8x8 도트 매트릭스	택트 스위치
						
택트 스위치 캡	볼 스위치	리드 스위치 센서	4x4키 매트릭스	5V 릴레이 모듈	조이스틱 모듈	수위 센서
						
온도센서 LM35	터미스터	습도센서	CDS 조도센서	불꽃감지센서	적외선 수신기	IR 리모컨

# 아두이노 키트(Kit) : Part-2

						
TCRT5000 적외선 센서	인체감지센서 모듈	사운드센서	능동부저	수동부저	초음파센서	I2C 1602 LCD 모듈
						
서보모터	스텝모터	스텝모터드라이버	RFID 수신 모듈	RFID 카드	RFID 태그	DS1302 RTC 모듈
						
1N4001 다이오드	2N2222 트랜지스터	74HC595	1x40 핀헤더	9V 배터리 스냅	아크릴 고정판	

■ 아두이노 UNO × 1	■ USB 케이블 × 1	■ 830핀브레드보드 × 1	■ 미니 브레드보드 × 1	■ 점퍼와이어세트 × 1
■ 듀폰케이블 × 80 (M/F,M/M)	■ 저항 × 30	■ 가변저항 × 1	■ LED × 20	■ RGB LED × 1
■ RGB LED 모듈 × 1	■ 1digit FND(CA) × 1	■ 4digit FND(CA) × 1	■ 8×8도트 매트릭스 × 1	■ 탭스위치 × 5
■ 탭스위치 캡 × 5	■ 볼스위치 × 1	■ 리드 스위치 센서 × 1	■ 4×4 키 매트릭스 × 1	■ 5V 릴레이 모듈 × 1
■ 조이스틱 모듈 × 1	■ 수위 센서 × 1	■ 온도센서 LM35 × 1	■ 써미스터 × 1	■ 온습도센서 × 1
■ CdS 조도센서 × 1	■ 불꽃감지센서 × 1	■ 적외선 수신기 × 1	■ IR 리모컨 × 1	■ TCRT5000 적외선 센서 × 1
■ 인체감지센서 모듈 × 1	■ 사운드센서 × 1	■ 능동부저 × 1	■ 수동부저 × 1	■ 초음파센서 × 1
■ I2C 1602 LCD 모듈 × 1	■ 서보모터 × 1	■ 스텝모터 × 1	■ 스텝모터드라이버 × 1	■ RFID 수신 모듈 × 1
■ RFID 카드 × 1	■ RFID 태그 × 1	■ DS1302 RTC 모듈 × 1	■ 1N4001 다이오드 × 1	■ 2N2222 트랜지스터 × 1
■ 74HC595 × 1	■ 1x40 핀헤더 × 1	■ 9V 배터리 스냅 × 1	■ 아크릴 고정판 × 1	