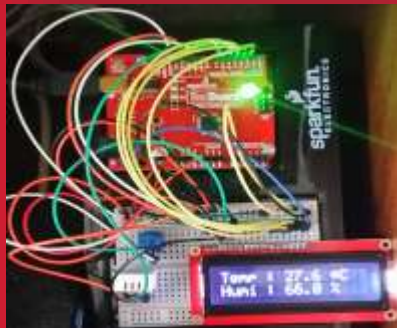




# Arduino-basic

## [wk13]

## various elements



Learn how to code Arduino from scratch

Comsi, INJE University

1<sup>st</sup> semester, 2023

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



# My ID (ARnn, github repo)

AR01	강동하
AR02	정재윤
AR03	유석진
AR04	정창민
AR05	정희서
AR06	유동기
AR07	장세진
AR08	정호기

위의 **id**를 이용해서 **github**에 **repo**를 만드시오.



# [Practice]

## ◆ [wk12]

- **Arduino : Motors & IRremote**
- **Complete your project**
- **Submit folder : ARnn\_Rpt11**

# wk12 : Practice-11 : ARnn\_Rpt11

## ◆ [Target of this week]

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

제출폴더명 : **ARnn\_Rpt11**

### 제출할 파일들

- ① **ARnn\_step\_motor.png**
- ② **ARnn\_step\_motor.ino**
- ③ **ARnn\_servo\_motor.ino**
- ④ **ARnn\_remote\_Serial.png**
- ⑤ **ARnn\_remote\_LCD.png**
- ⑥ **ARnn\_remote\_LCD.ino**
- ⑦ **\*.ino**



# 9. Various elements

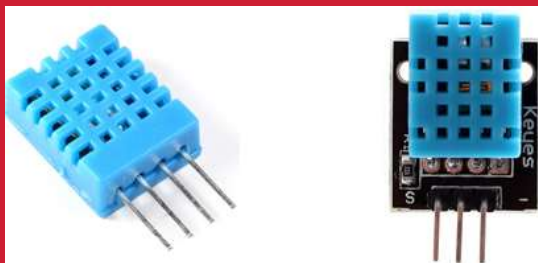
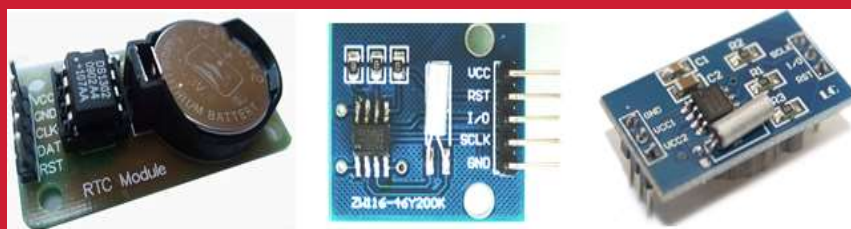


그림 9. 2 DHT11과 DHT11 모듈



# 9. 여러가지 부품들

- 9.1 버저
- 9.2 온습도 센서
- 9.3 실시간 클럭모듈
- 9.4 RFID
- 9.5 초음파 거리센서



# 9.1 버저



마그네틱 버저,

피에조 버저,

기계식 버저

## 버저(Buzzer)



그림 9. 1 마그네틱 버저, 피에조 버저, 기계식 버저

- ✓ 전기적 신호로 진동판을 진동시켜 소리를 출력하는 부품
- ✓ 마그네틱 버저, 피에조 버저, 기계식 버저 등이 있음
- ✓ 피에조 버저는 일정 주파수를 입력시켜 다양한 음을 낼 수 있음 (수동 버저)

<https://devicemart.blogspot.com/2019/05/blog-post.html>

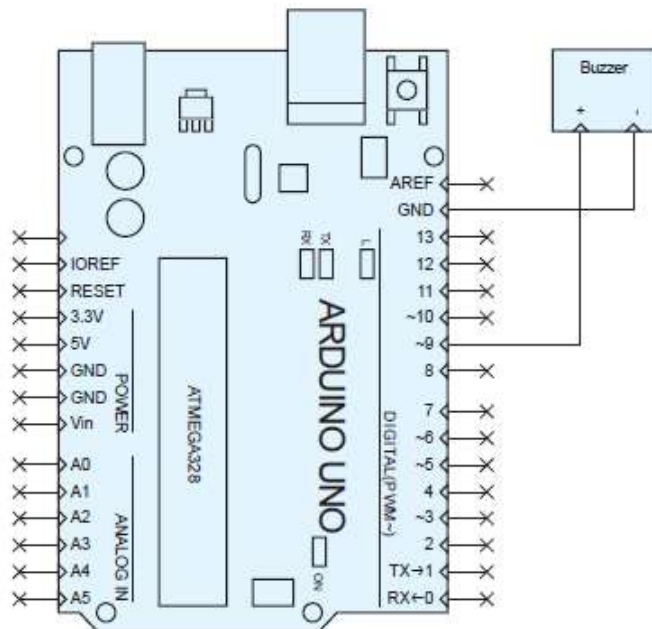


## EX 9.1

## 피에조 버저를 이용한 소리 출력 (1/3)

**실습목표** 피에조 버저를 이용하여 다양한 소리를 출력한다.

- Hardware**
1. 버저의 (+)핀을 Arduino의 9번핀에 연결한다.
  2. 버저의 (-)핀을 Arduino의 GND에 연결한다.



EX 9.1

## 피에조 버저를 이용한 소리 출력 (2/3)

**Commands**    • pinMode(핀번호, 설정)

핀의 입출력 모드를 설정한다. '핀번호' 에는 설정하고자 하는 핀의 번호와 '설정'에는 입력으로 사용하기 위해선 'INPUT', 출력으로 사용하기 위해선 'OUTPUT', 입력이며 풀업 사용시 'INPUT\_PULLUP'을 적는다.

• for(변수=시작 값 ; 조건 ; 변수의 증분){ }

변수의 시작 값부터 조건이 만족하는 경우 '{ }' 내의 명령을 수행한다. '변수의 증분'에서는 1회 명령이 수행될 때 마다 변수를 증가 혹은 감소시킨다.

• **tone**(핀 번호, 주파수, 시간)

해당 주파수와 50%의 듀티비의 사각파를 핀에 출력한다. 시간은 밀리초 단위로 설정할 수 있다.

## 9.1.3 피에조 버저

EX 9.1

피에조 버저를 이용한 소리 출력 (3/3)

- Sketch 구성**
1. 버저를 디지털 입출력핀 9번으로 설정한다.
  2. '도레미파솔라시도' 음에 대하여 피에조 버저의 진동 주파수를 설정한다.
  3. 도레미송 악보를 데이터화하여 시간에 맞춰 해당 주파수로 피에조 버저를 진동시킨다.

**실습 결과** 도레미송이 반복하여 연주된다.

ex\_9\_1\_start

```

1  /*
2  예제 9.1
3  피에조 버저를 이용한 소리 출력
4  */
5
6  int buzzerPin = 9;
7  int songLength = 16;
8
9  // 노래 데이터, 공백은 쉬는 구간을 나타낸다
10 char notes[] = "cee egg dff abb ";
11 // 음의 길이, 노래 데이터와 맞춰 음의 길이를 설정한다.
12 int beats[] = {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1};
13
14 // 노래의 빠르기를 설정한다.
15 int tempo = 200;
16
17 void setup()
18 {
19   // 부저핀을 출력으로 설정한다
20   pinMode(buzzerPin, OUTPUT);
21 }

```

```

24 void loop()
25 {
26   // 부저 출력 시간에 사용할 변수 설정
27   int duration;
28
29   // 노래 길이 데이터 갯수만큼 반복한다
30   for (int i = 0; i < songLength; i++){
31     // 한 음의 시간을 계산한다.
32     duration = beats[i] * tempo;
33
34     if (notes[i] == ' '){      // 공란일 경우 음을 출력하지 않는다
35       delay(duration);
36     }
37     else{
38       // tone 명령어를 통하여 부저 핀으로 사각파를 출력한다
39       tone(buzzerPin, frequency(notes[i]), duration);
40       delay(duration);
41     }
42     // 음이 바뀔 때 잠시 쉬어준다
43     delay(tempo / 10);
44   }
45 }

```

## 9.1.4 피에조 버저: code-2

```

47 int frequency(char note){
48     // 노래 데이터를 주파수 값으로 변경하기 위한 함수
49
50     int i;
51     // 음계의 갯수 설정
52     int notes = 8;
53
54     char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };
55     int frequencies[] = {262, 294, 330, 349, 392, 440, 494, 523};
56
57     // 노래 데이터를 주파수 값으로 변경하기 위해 반복하여 비교한다
58     for (i = 0; i < notes; i++){
59         if (names[i] == note){
60             // 맞는 값을 찾았을 경우 이 값을 회신한다
61             return(frequencies[i]);
62         };
63     };
64     // 앞의 for문에서 맞는 값을 못찾았을 경우 0을 회신한다
65     return(0);
66 }

```

DIY

응용 문제

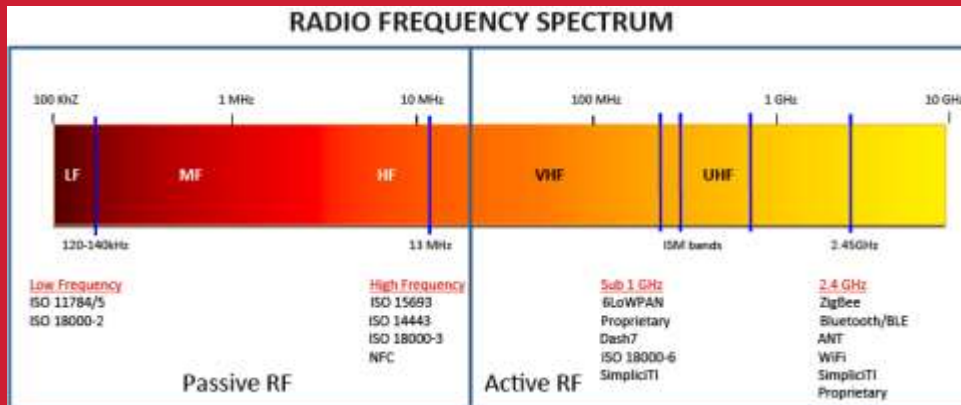
세 개의 스위치 입력을 받아 각 스위치가 ‘도’, ‘미’, ‘솔’ 음을 내어 연주할 수 있는 스케치를 만들어 보자.

아두이노 스케치 코드를  
**ARnn\_doremi.ino** 로 저장...



# 9.4

# RFID



[source](#)

[RFID 주파수와 용도](#)

## RFID (Radio-Frequency Identification)

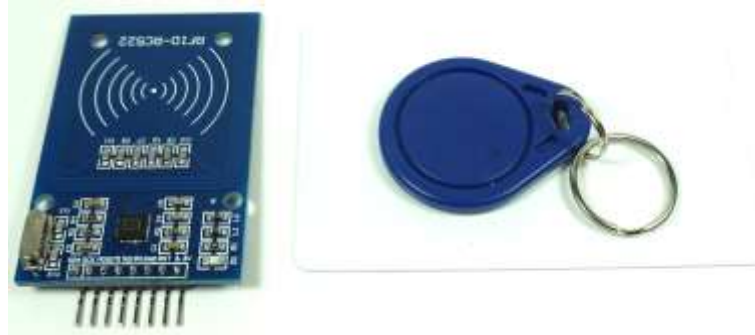


그림 9. 5 실험에 사용할 RC522 RFID 모듈과 태그

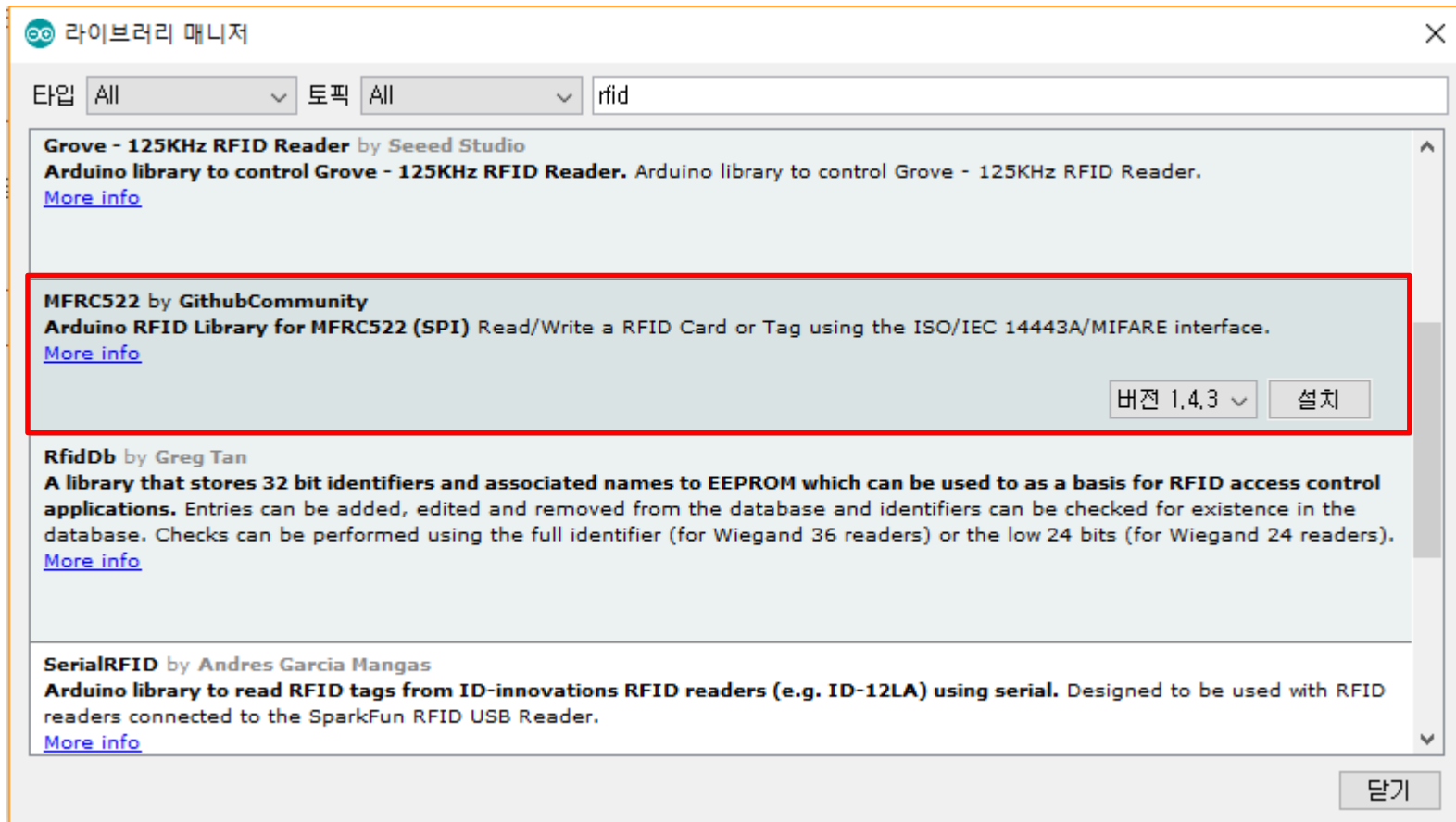
- ✓ 전파를 이용하여 원거리의 정보를 인식하는 기술 (비접촉)
- ✓ RFID 태그(RFID tag)와 RFID 판독기(RFID reader)로 구성
- ✓ RFID 태그에는 안테나와 직렬회로가 내장되어 있어 RFID 판독기에 접근하였을 때 무선통신으로 데이터 송수신
- ✓ RFID 태그의 전원 유무에 따라 수동형, 반수동형, 능동형으로 구분
- ✓ SPI 통신을 통해 MRFC522 IC를 이용한 RFID 판독기 모듈과 통신

※ 하기의 주소에서 라이브러리를 다운받아 설치 할 것  
<https://github.com/miguelbalboa/rfid>

수동형, 능동형 RFID



✓ 라이브러리 관리' 메뉴에서 라이브러리 매니저를 실행 시켜 RFID 라이브러리를 설치하자.



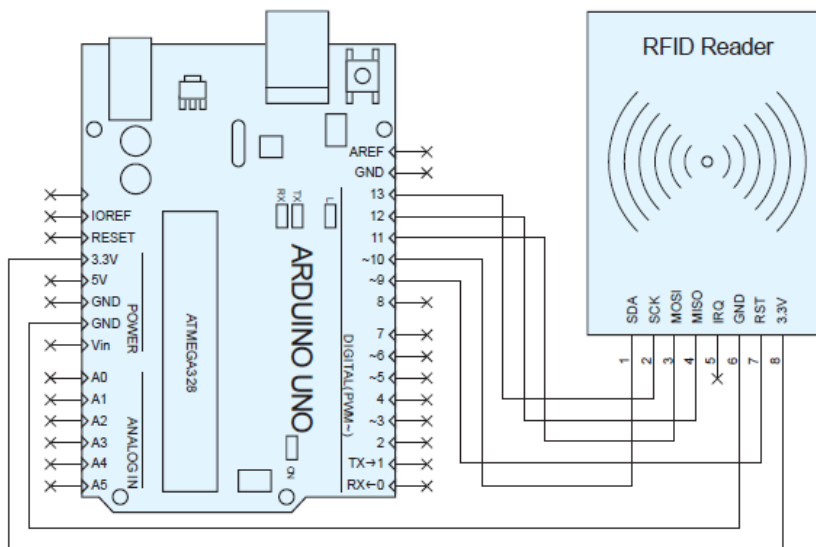
## EX 9.4

## RFID (1/3)

- 실습목표**
1. RFID 태그를 읽는다.
  2. RFID 태그의 UID와 PICC type을 판독하여 시리얼 통신으로 출력한다.

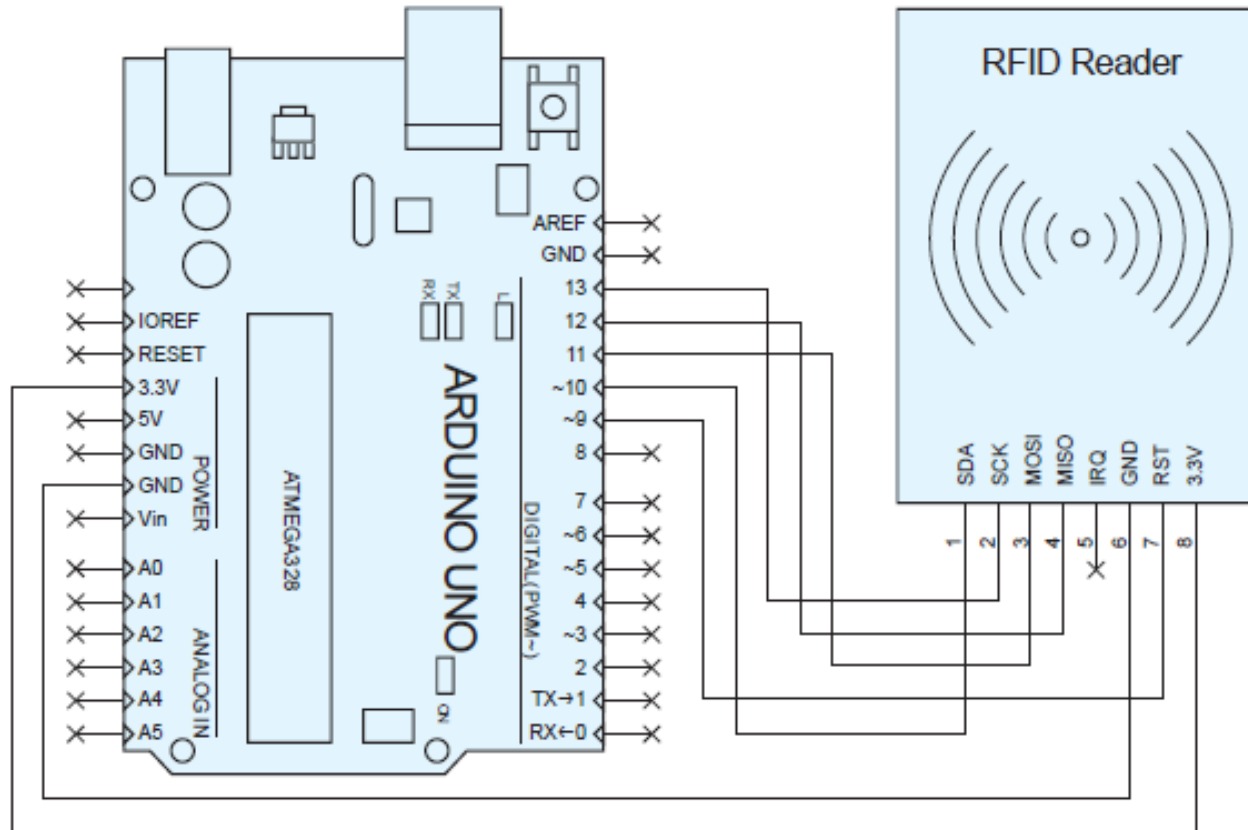
(UID : Unique Identifier, PICC: Proximity Integrated Circuit Card)

- Hardware**
1. RFID 리더와 Arduino는 SPI 통신으로 데이터를 주고 받는다.
  2. RFID 리더의 3.3V, GND 핀을 Arduino의 3.3V, GND에 각각 연결한다.
  3. RFID 리더의 SDA, SCK, MOSI, MISO, RST 핀을 Arduino의 10, 13, 11, 12, 9번 핀에 연결한다.



EX 9.4

RFID (1/3)



- RFID 리더의 3.3V, GND 핀을 Arduino의 3.3V, GND에 연결한다.
- RFID 리더의 SDA, SCK, MOSI, MISO, RST 핀을 Arduino의 10, 13, 11, 12, 9번 핀에 연결한다

## EX 9.4

## RFID (2/3)

**Commands**

- MFRC522 mfrc522(SS\_PIN, RST\_PIN)  
'mfrc522'란 이름으로 RFID 리더를 설정한다.
- mfrc522.PICC\_IsNewCardPresent  
'mfrc522'란 이름의 RFID 리더에 새로운 카드가 입력되었을 때 'TRUE' 값을 출력한다.
- mfrc522.PICC\_ReadCardSerial()  
'mfrc522'란 이름의 RFID 리더에서 카드의 내용을 읽는다.
- mfrc522.uid.uidByte[ ]  
'mfrc522'란 이름의 RFID 리더에서 읽어낸 데이터 중 uid 값에 데이터
- mfrc522.PICC\_GetType(mfrc522.uid.sak)  
'mfrc522'란 이름의 RFID 리더에서 읽어낸 데이터 중 PICC 데이터를 읽는다.
- mfrc522.PICC\_GetTypeName(picctype)  
'mfrc522'란 이름의 RFID 리더에서 읽어낸 PICC 이름 데이터.
- mfrc522.PICC\_HaltA()  
'mfrc522'란 이름의 RFID 리더를 중단한다.
- mfrc522.PCD\_StopCrypto1()  
'mfrc522'란 이름의 RFID 리더를 초기화 한다.

## EX 9.4

## RFID (3/3)

- Sketch 구성**
1. 'mfrc522'란 이름으로 RFID 리더를 설정한다.
  2. 새로운 RFID 입력이 있을 경우 데이터를 읽는다.
  3. UID와 PICC를 읽는다.
  4. UID와 PICC를 시리얼 통신으로 출력한다.
  5. RFID 리더를 초기화 하고 새로운 카드 수신을 대기한다.

**실습 결과** RFID 태그를 판독기에 접근시킬 때 UID와 PICC type이 출력된다.

```
COM11 (Arduino/Genuino Uno)

Please touch your card

Card UID: 86 17 C5 1F   PICC type: MIFARE 1KB

Card UID: D9 F6 B6 C3   PICC type: MIFARE 1KB
```

ex\_9\_4

```

1 /*
2 예제 9.4
3 RFID
4 */
5
6 // SPI 라이브러리를 불러온다.
7 #include <SPI.h>
8 // MFRC522 라이브러리를 불러온다
9 #include <MFRC522.h>
10
11 // SS 핀을 10번 핀으로 설정한다.
12 #define SS_PIN 10 //Arduino Uno, Data
13 // Reset 핀을 9번 핀으로 설정한다.
14 #define RST_PIN 9
15
16 // mfrc522란 이름의 RFID 판독기를 설정한다.
17 MFRC522 mfrc522(SS_PIN, RST_PIN);
18
19 void setup() {
20 // 시리얼 통신을 설정한다.
21 Serial.begin(9600);
22 // SPI 통신을 시작한다.
23 SPI.begin();
24 // 앞서 설정한 mfrc522란 이름의 RFID 판독기를 시작한다.
25 mfrc522.PCD_Init(); // Init MFRC522 card
26 Serial.println("Please touch your card");
27 Serial.println(" ");
28 }

```

```

30 void loop() {
31 // 새로운 카드를 기다린다.
32 if (!mfrc522.PICC_IsNewCardPresent()) return;
33 // 카드를 읽는다.
34 if (!mfrc522.PICC_ReadCardSerial()) return;
35 // 'Card UID: '메세지를 출력한다.
36 Serial.print("Card UID:");
37
38 // 판독기에 입력된 UID의 바이트수 만큼 읽어낸다.
39 // UID는 카드의 종류에 따라 최대 8바이트를 갖는다.
40 for (byte i = 0; i < mfrc522.uid.size; i++) {
41 // UID를 시리얼 통신으로 출력한다.
42 Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
43 Serial.print(mfrc522.uid.uidByte[i], HEX);
44 }
45
46 // 시리얼 통신으로 'PICC type: '메세지를 출력한다.
47 Serial.print(" PICC type: ");
48 // picType 변수에 picc type을 저장한다.
49 byte piccType = mfrc522.PICC_GetType(mfrc522.uid.sak);
50 // 시리얼 통신으로 picc type을 출력한다.
51 Serial.println(mfrc522.PICC_GetTypeName(piccType));
52 // 줄바꿈
53 Serial.println(" ");
54
55 // mfrc522 판독기를 초기화 한다
56 mfrc522.PICC_HaltA(); // Halt PICC
57 mfrc522.PCD_StopCrypto1(); // Stop encryption on PCD
58 }

```

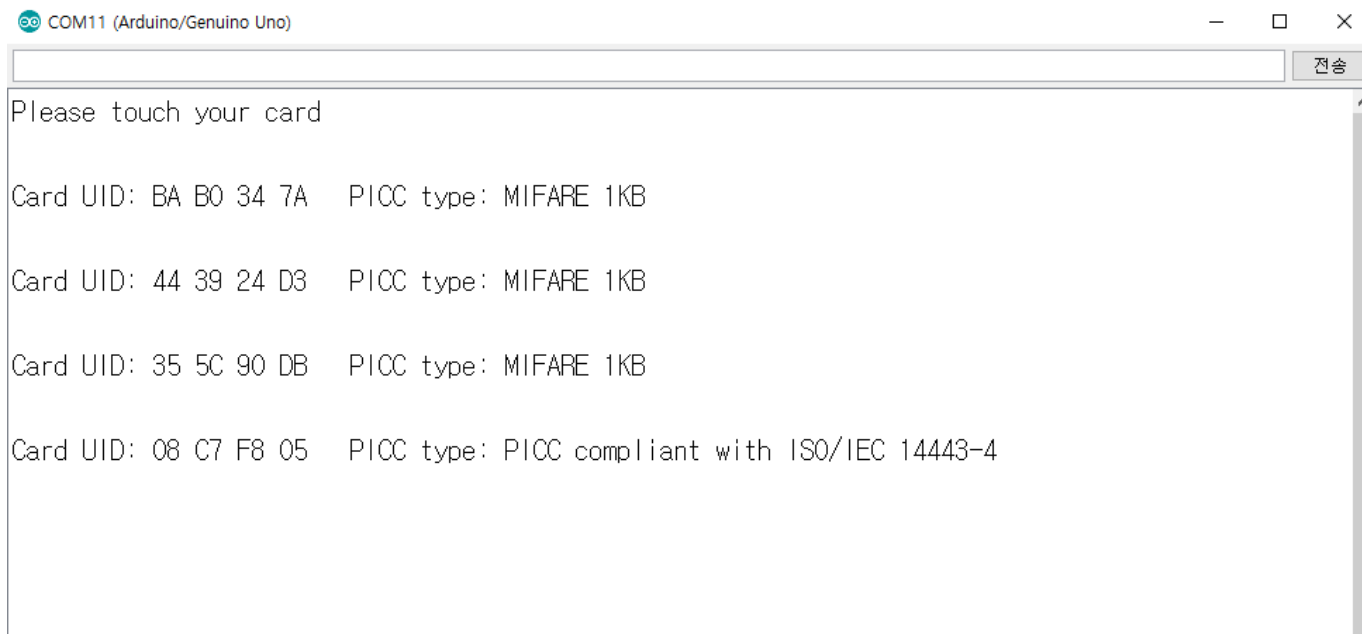
## DIY

1. 갖고 있는 교통카드를 판독기에 접근시켜 읽어보자.

## 응용 문제

2. 폰의 NFC를 활성화 시킨 후 uid와 PICC type을 출력해보자.

직렬모니터 출력 화면을  
**ARnn\_rfid.png** 로 저장...



COM11 (Arduino/Genuino Uno)

Please touch your card

Card UID: BA B0 34 7A    PICC type: MIFARE 1KB

Card UID: 44 39 24 D3    PICC type: MIFARE 1KB

Card UID: 35 5C 90 DB    PICC type: MIFARE 1KB

Card UID: 08 C7 F8 05    PICC type: PICC compliant with ISO/IEC 14443-4



# 9.5

## 초음파 거리센서





# 9.5 초음파 거리센서 (HC-SR04)

## 초음파 거리센서 (HC-SR04)



그림 9.6 실험에 사용할 HC-SR04 초음파센서

- ✓ 약 40 kHz의 주파수의 초음파를 발사하여 물체에 반사되어 돌아오는 시간을 측정
- ✓ 외부 환경에 강한 특징을 갖고 있고, 물체의 색깔에 상관없이 사용할 수 있으며, 투명한 물체도 감지 가능하며 물이나 먼지 등이 있더라도 감지할 수 있는 장점이 있음
- ✓ 외부에 초음파 발신부가 노출되어야 함

$$L = \frac{\text{에코 펄스 폭} * 340[m/s]}{2}, L : \text{물체와의 거리}$$

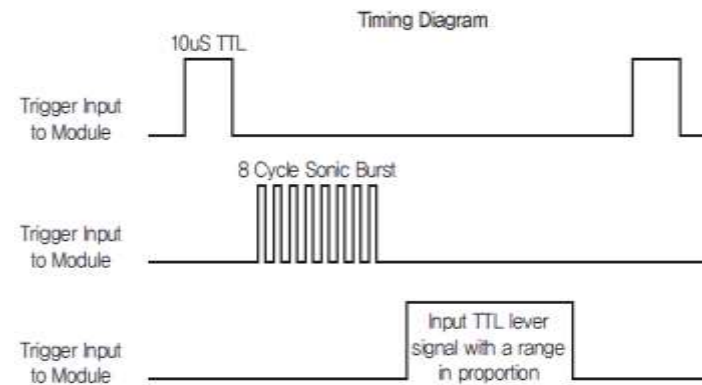


그림 9.7 HC-SR04의 타이밍 다이어그램

표 9.1 HC-SR04 사양표

동작 전압	DC 5V
소비 전력	13mA
동작 주파수	40kHz
최대 감지거리	4m
최소 감지거리	2cm
측정 각도	15°
트리거 입력 신호	10us TTL 펄스
에코 출력 신호	거리에 따른 TTL 레벨의 신호
사이즈	45*20*15mm

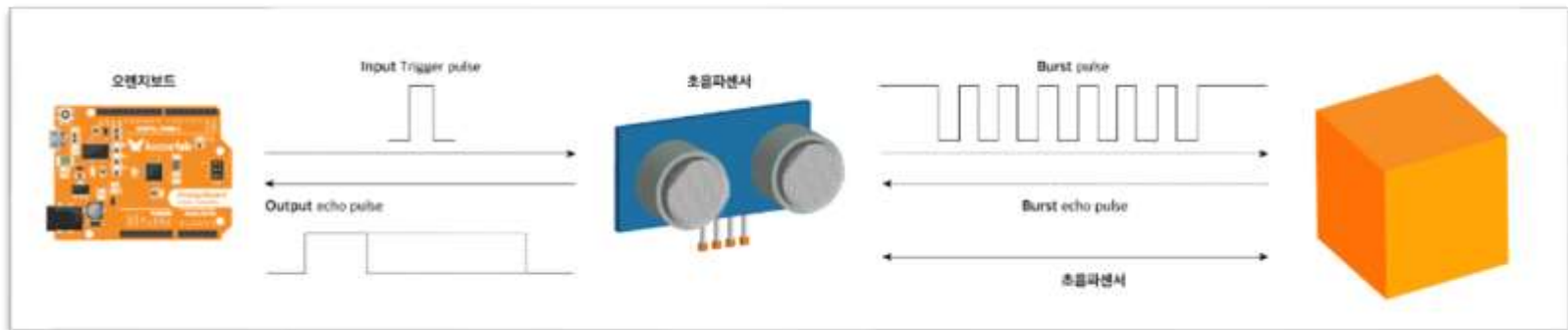
## 9.5 초음파 거리센서 (HC-SR04)

$$t = \frac{2 \times L(\text{물체와의 거리}m)}{V_s(\text{음속}m/s)}$$

t: 신호가 되돌아 올때까지 걸리는 시간(s)

$t = 2 * 0.01 / 340 = 58.824\mu s$  로, 초음파가 **1cm**를 이동하는데 걸리는 시간은 약 **29 $\mu s$** 가 걸리며, 초음파가 반사된 물체와의 거리는 다음과 같이 구할 수 있습니다.

측정 거리 (cm) = duration (양복에 걸린 시간) / 29 / 2 (양복)  
= duration (양복에 걸린 시간) / 58



<https://kocoafab.cc/tutorial/view/357>

# 9.5.1 초음파 거리센서 (HC-SR04)

EX 9.5

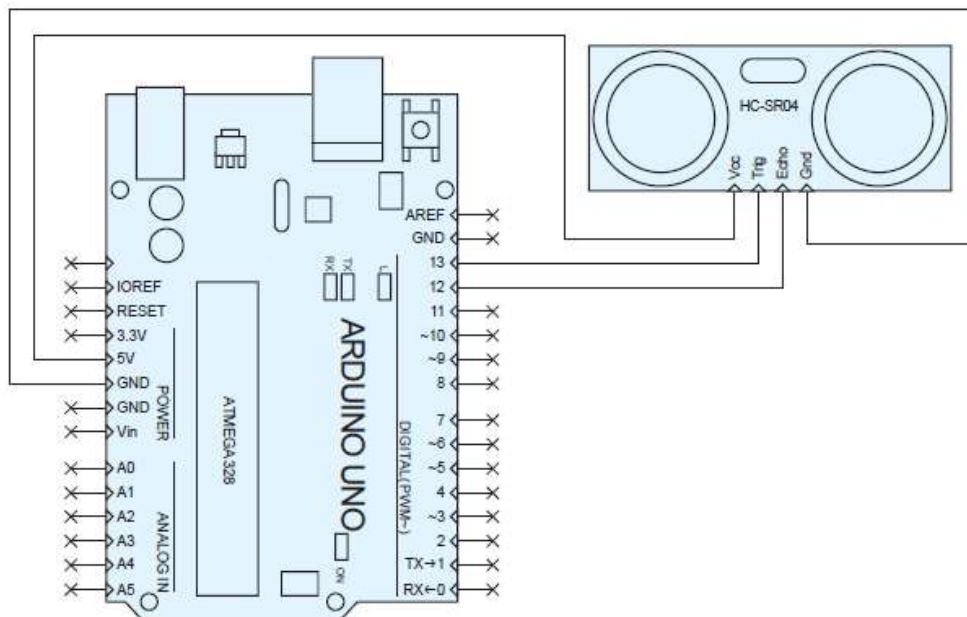
초음파 거리센서를 이용한 거리 측정 (1/3)

**실습목표** 1. 초음파 모듈 센서를 이용하여 거리를 측정한다.

2. 측정한 거리의 변화가 있을 때 시리얼 통신을 이용하여 모니터에 출력한다.

**Hardware** 1. HC-SR04 모듈의 Vcc와 GND를 Arduino의 5V와 GND에 연결한다.

2. HC-SR04 모듈의 Echo핀과 Trig핀을 Arduino의 12, 13번핀에 연결한다.



EX 9.5

초음파 거리센서를 이용한 거리 측정 (2/3)

### Commands

- `pulseIn`(핀번호, 값, 초과시간)

핀번호로 입력된 펄스에 대하여 펄스의 폭을 마이크로초( $\mu$ s) 단위로 측정한다. '핀번호'엔 펄스를 입력받을 핀의 번호, '값'엔 HIGH 펄스폭을 측정할 때는 HIGH, LOW 펄스폭을 측정할 때는 LOW를 적는다. 초과시간은 최대 측정 시간으로서 초기값은 1이다.

- `delayMicroseconds`(지연시간)

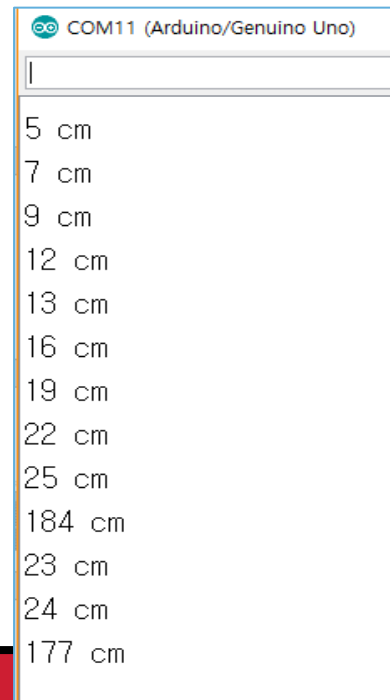
지연시간에는 잠시 동작을 지연시키기 위한 값을 넣는다. 마이크로초 단위로 넣는다.

EX 9.5

초음파 거리센서를 이용한 거리 측정 (3/3)

- Sketch 구성**
1. 디지털 입출력핀 12와 13을 각각 에코핀과 트리거핀으로 설정한다.
  2. 폭 10  $\mu$ s의 펄스를 트리거 핀으로 출력한다.
  3. 'pulseIn' 명령어를 이용하여 에코핀으로 입력되는 HIGH 펄스의 폭을 측정한다.
  4. 펄스폭에 대하여 초음파의 속도를 물체와의 거리를 측정한다.
  5. 현재 측정한 거리와 이전에 측정한 거리가 상이할 때 그 값을 시리얼 통신으로 출력한다.

**실행 결과** 초음파 센서 앞의 물체와의 거리에 따라 측정값이 시리얼 모니터에 출력된다.  
거리의 변화가 없을 때는 출력하지 않는다.



ex\_9\_5\_start

```

1 /*
2  예제 6.7
3  초음파 거리센서를 이용한 거리 측정
4 */
5
6 // 트리거 핀과 에코 핀 번호를 설정한다.
7 const char trigPin = 13;
8 const char echoPin = 12;
9
10 // 펄스 폭과 거리 변수 설정
11 int pulseWidth;
12 int distance;
13 int distanceOld;
14
15 void setup() {
16   // 시리얼 통신 설정
17   Serial.begin(9600);
18   // 트리거 핀은 출력으로, 에코핀은 입력으로 설정
19   pinMode(trigPin, OUTPUT);
20   pinMode(echoPin, INPUT);
21   // 트리거 핀의 초기값을 LOW로 한다
22   digitalWrite(trigPin, LOW);
23 }

```

```

25 void loop(){
26   // 10us의 트리거 신호를 HC-SR04로 내보낸다.
27   digitalWrite(trigPin, HIGH);
28   delayMicroseconds(10);
29   digitalWrite(trigPin, LOW);
30
31   // Echo 펄스 폭을 측정하여 pulseWidth 변수에 저장한다.
32   pulseWidth = pulseIn(echoPin, HIGH);
33   // 거리를 계산한다.
34   distance = pulseWidth / 58;
35
36   // 감지거리인 2~200cm 범위의 거리값만 사용한다.
37   if(distance <= 200 || distance >= 2){
38     // 이전의 거리값과 비교하여 변화가 있을 경우에만
39     // 시리얼 통신으로 전송한다.
40     if(distance != distanceOld){
41       Serial.print(distance);
42       Serial.println(" cm");
43     };
44   };
45   distanceOld = distance;
46   delay(100);
47 }

```

### DIY

### 응용 문제

1. 황색, 청색, 적색 LED를 Arduino에 연결하자.
2. 물체와의 거리가 2~30cm에서는 황색 LED, 31~60cm에서는 청색 LED, 그 이상의 거리에서는 적색 LED가 점등 되게 스케치를 작성하여라.

동작 중 사진 또는 동영상을

**ARnn\_ultrasonic.png**로 저장하고 제출,

어두이노 스케치 코드를

**ARnn\_ultrasonic.ino**로 제출



# [Practice]

## ◆ [wk13]

- **Arduino : various elements**
- **Complete your project**
- **Submit folder : ARnn\_Rpt12**



# wk13 : Practice-12 : ARnn\_Rpt12

## ◆ [Target of this week]

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

제출폴더명 : **ARnn\_rpt12**

### 제출할 파일들

- ① **ARnn\_doremi.ino**
- ② **ARnn\_rfid.png**
- ③ **ARnn\_ultrasonic.png**
- ④ **ARnn\_ultrasonic.ino**
- ⑤ **\*.ino**

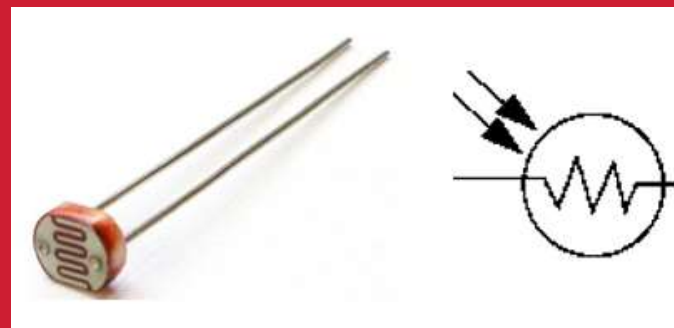
# wk15 : 기말고사 안내

## 아두이노 실기

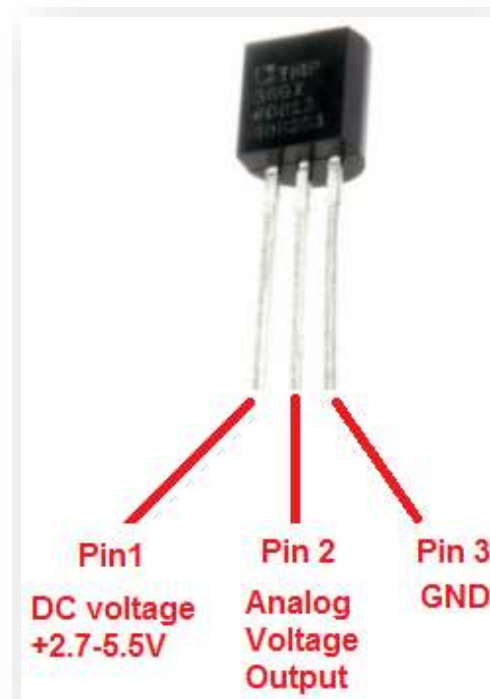
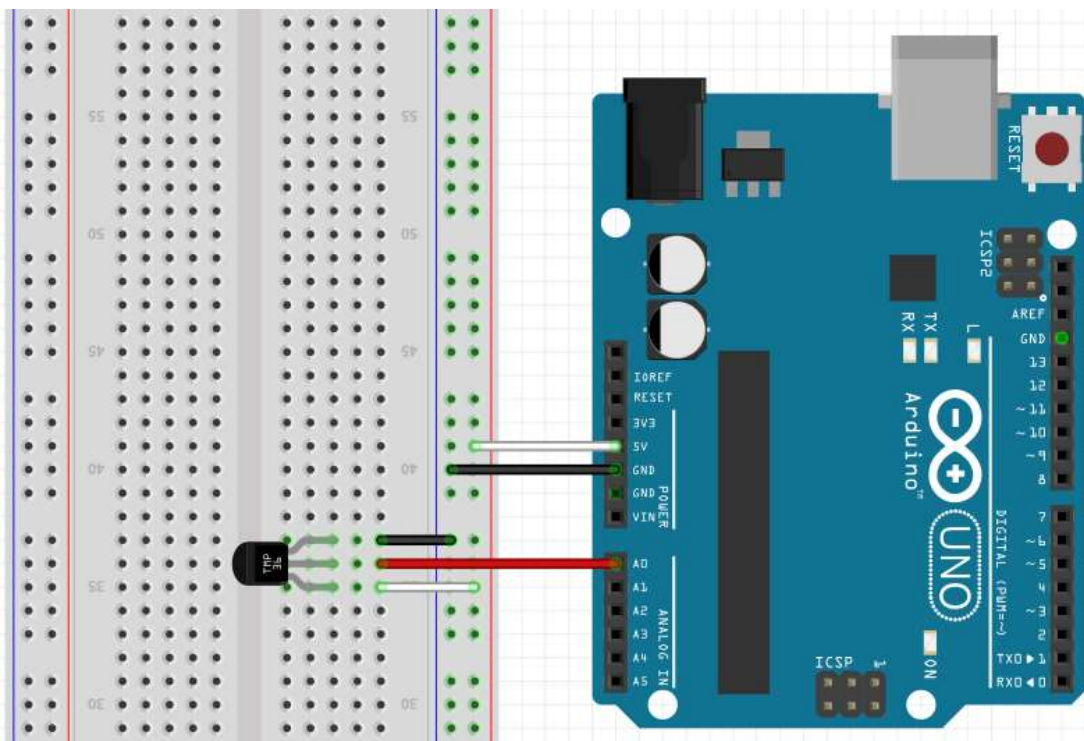
- 시간: 6월 15일 오후 2 시~
- 장소: E323 실습실
- 배점: @@@
- 각자 **github**에 **project** 폴더로 업!



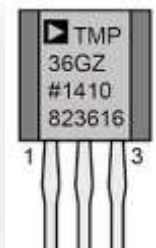
# 온도, 조도 센서



# A3.1.1 Temperature sensor [ TMP36]



## Parts : TMP36



- **Size:** TO-92 package (about 0.2" x 0.2" x 0.2") with three leads
- **Price:** \$2.00 at the [Adafruit shop](#)
- **Temperature range:** -40°C to 150°C / -40°F to 302°F
- **Output range:** 0.1V (-40°C) to 2.0V (150°C) but accuracy decreases after 125°C
- **Power supply:** 2.7V to 5.5V only, 0.05 mA current draw

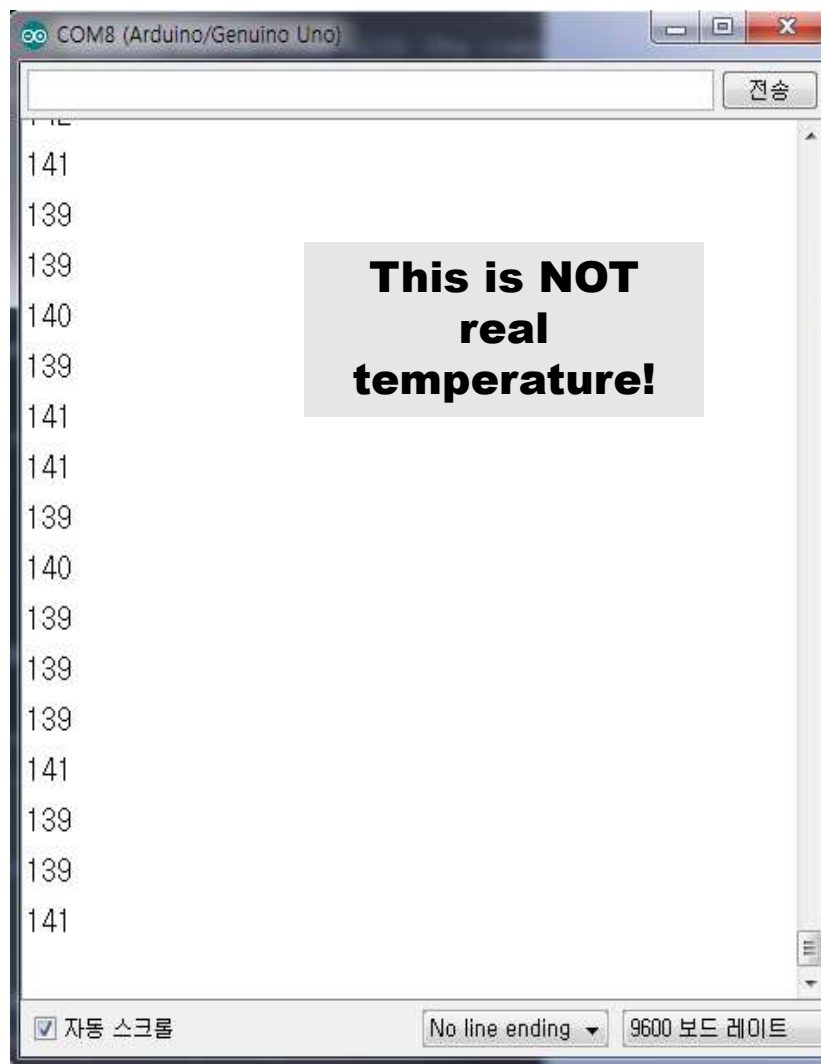


## A3.1.2 Temperature sensor [ TMP36 ]

### Simple code

```
TMP36 $
1 //
2 //  AR00, TMP36 sensor
3 //
4
5 #define TEMP_INPUT 0
6 // or  int TEMP_INPUT = 0;
7
8 void setup() {
9   Serial.begin(9600);
10 }
11
12 void loop() {
13
14   int value = analogRead(TEMP_INPUT);
15   Serial.println(value);
16
17   delay(1000);
18 }
```

### Serial output (0 ~ 1023)



# A3.1.3 Temperature sensor [ TMP36]

## Sensor property

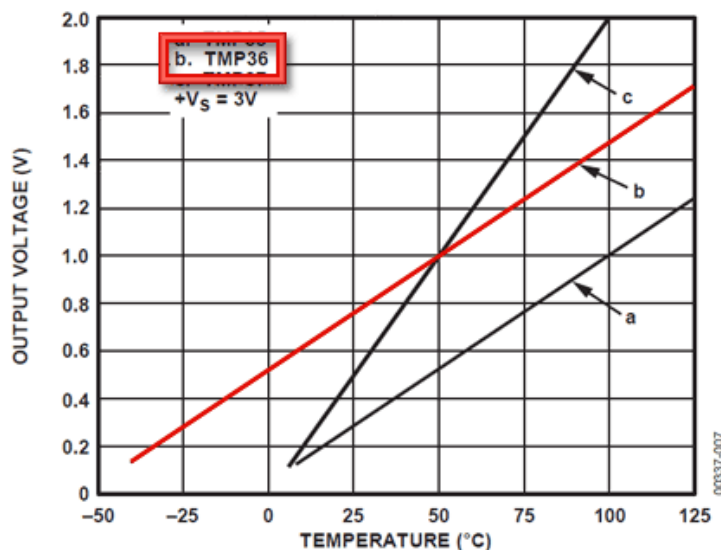


Figure 6. Output Voltage vs. Temperature

### Output Voltage (mV) vs. Temperature (°C)

V	0	500	1000
T	-50	0	50

[https://github.com/Redwoods/Arduino/blob/master/ar-iot/py-ml/tmp36\\_LR.ipynb](https://github.com/Redwoods/Arduino/blob/master/ar-iot/py-ml/tmp36_LR.ipynb)

## Temperature conversion

$$\text{Temp (}^{\circ}\text{C)} = (\text{Vout} - 500) / 10$$

$$\text{Vout (mV)} = \text{value} * (5000 / 1023)$$

$$(0 \leq \text{value} \leq 1023)$$



```
// converting that reading to voltage
float voltage = value * 5.0 * 1000; // in mV
voltage /= 1023.0;
float temperatureC = (voltage - 500) / 10 ;
```



## A3.1.4 Temperature sensor [ TMP36 ]

### Working code

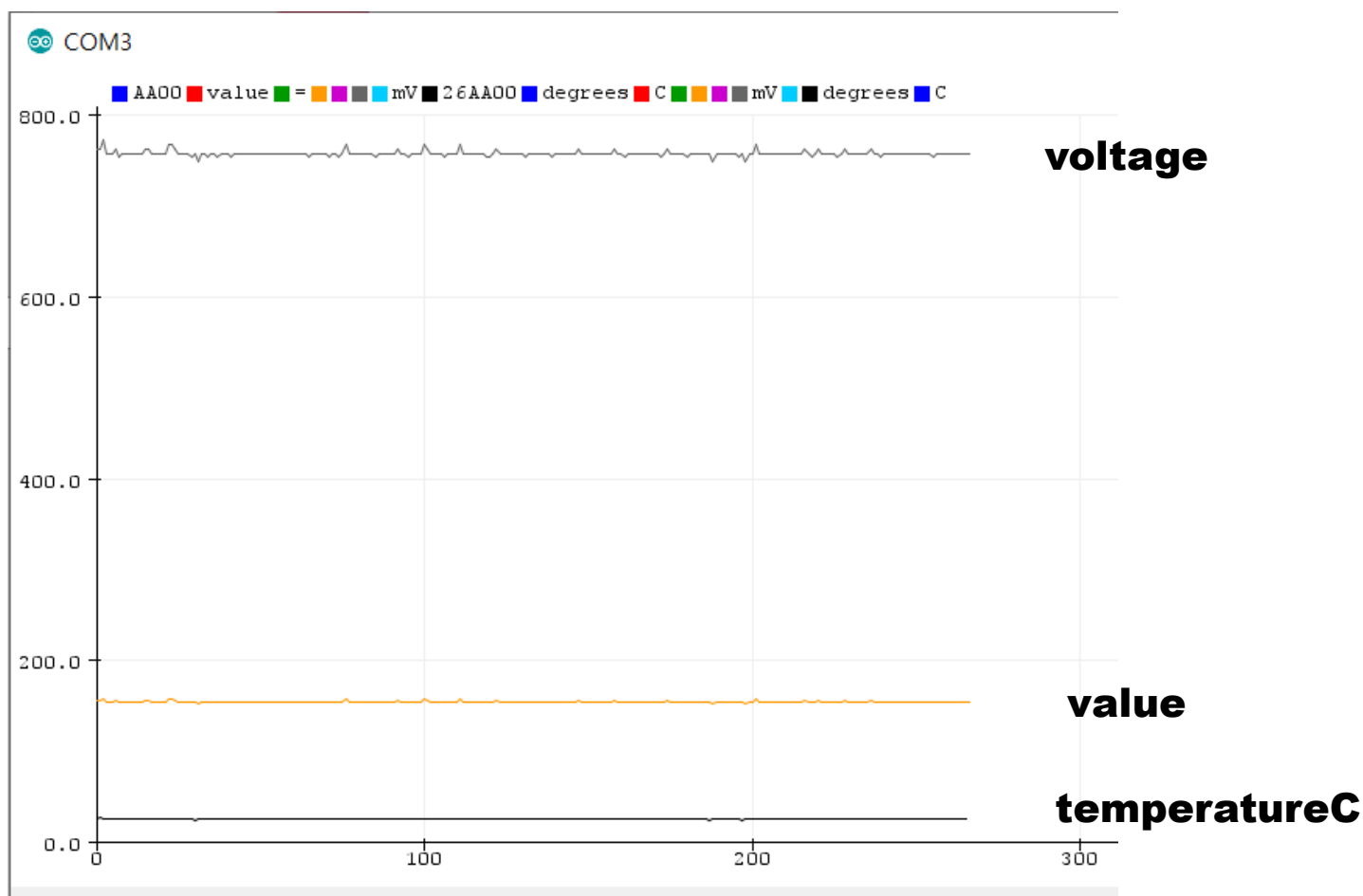
```
TMP36
10 }
11
12 void loop() {
13     //getting the voltage reading from the temperature sensor
14     int value = analogRead(TEMP_INPUT);
15     Serial.print("AA00, value = ");
16     Serial.print(value);
17     Serial.print(" : ");
18
19     // converting that reading to voltage
20     float voltage = value * 5.0 * 1000; // in mV
21     voltage /= 1023.0;
22
23     // print out the voltage
24     Serial.print(voltage);
25     Serial.print(" mV, ");
26
27     // now print out the temperature
28     float temperatureC = (voltage - 500) / 10 ;
29     Serial.print(temperatureC);
30     Serial.println(" degrees C");
31
32     delay(1000);
33 }
```

### Serial output ( °C )

```
COM4
AA00, value = 131 : 640.27 mV, 14.03 degrees C
AA00, value = 130 : 635.39 mV, 13.54 degrees C
AA00, value = 132 : 645.16 mV, 14.52 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 129 : 630.50 mV, 13.05 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 130 : 635.39 mV, 13.54 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 132 : 645.16 mV, 14.52 degrees C
AA00, value = 129 : 630.50 mV, 13.05 degrees C
AA00, value = 132 : 645.16 mV, 14.52 degrees C
AA00, value = 129 : 630.50 mV, 13.05 degrees C
AA00, value = 130 : 635.39 mV, 13.54 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
AA00, value = 128 : 625.61 mV, 12.56 degrees C
```

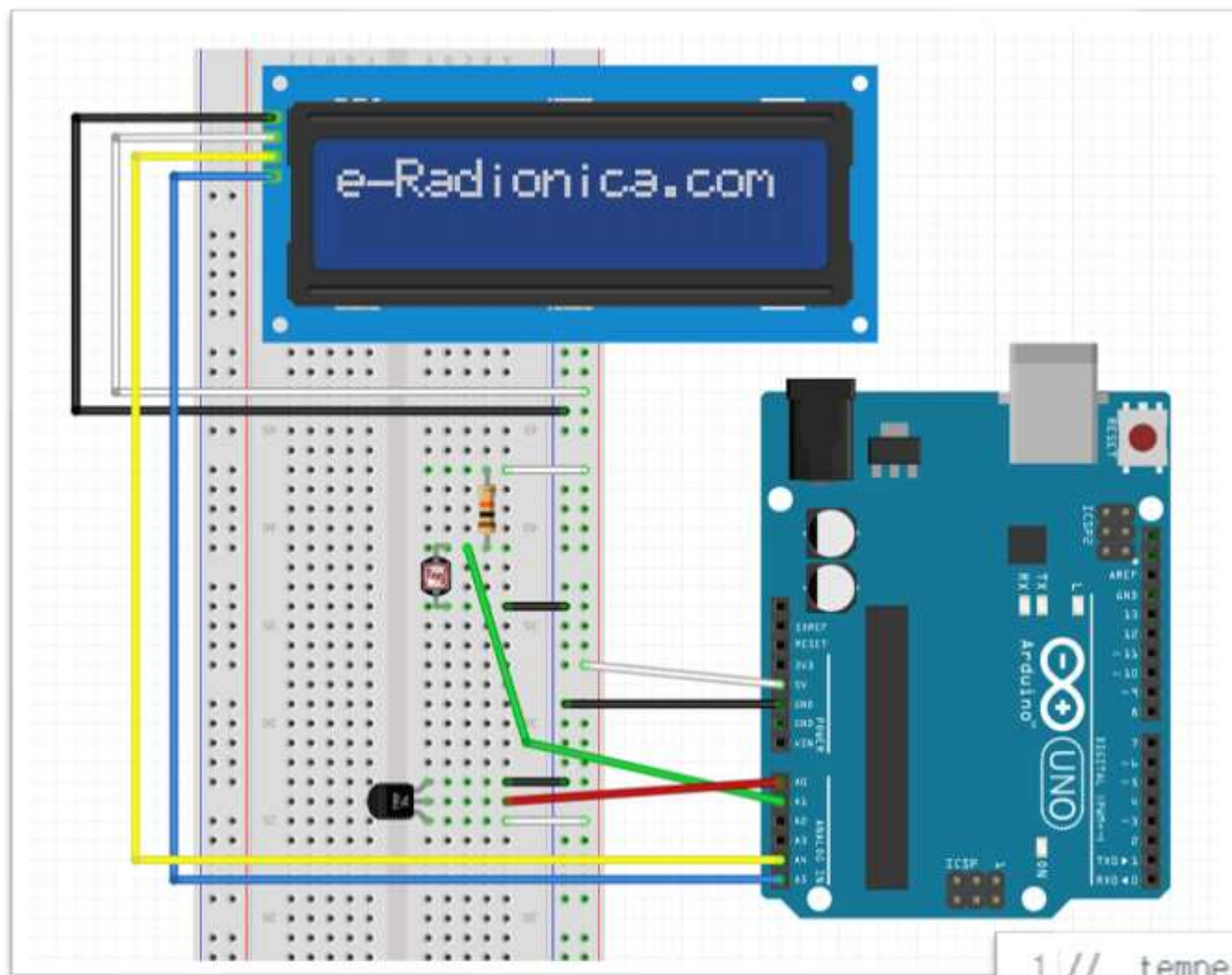


# A3.1.5 Temperature sensor [ TMP36]





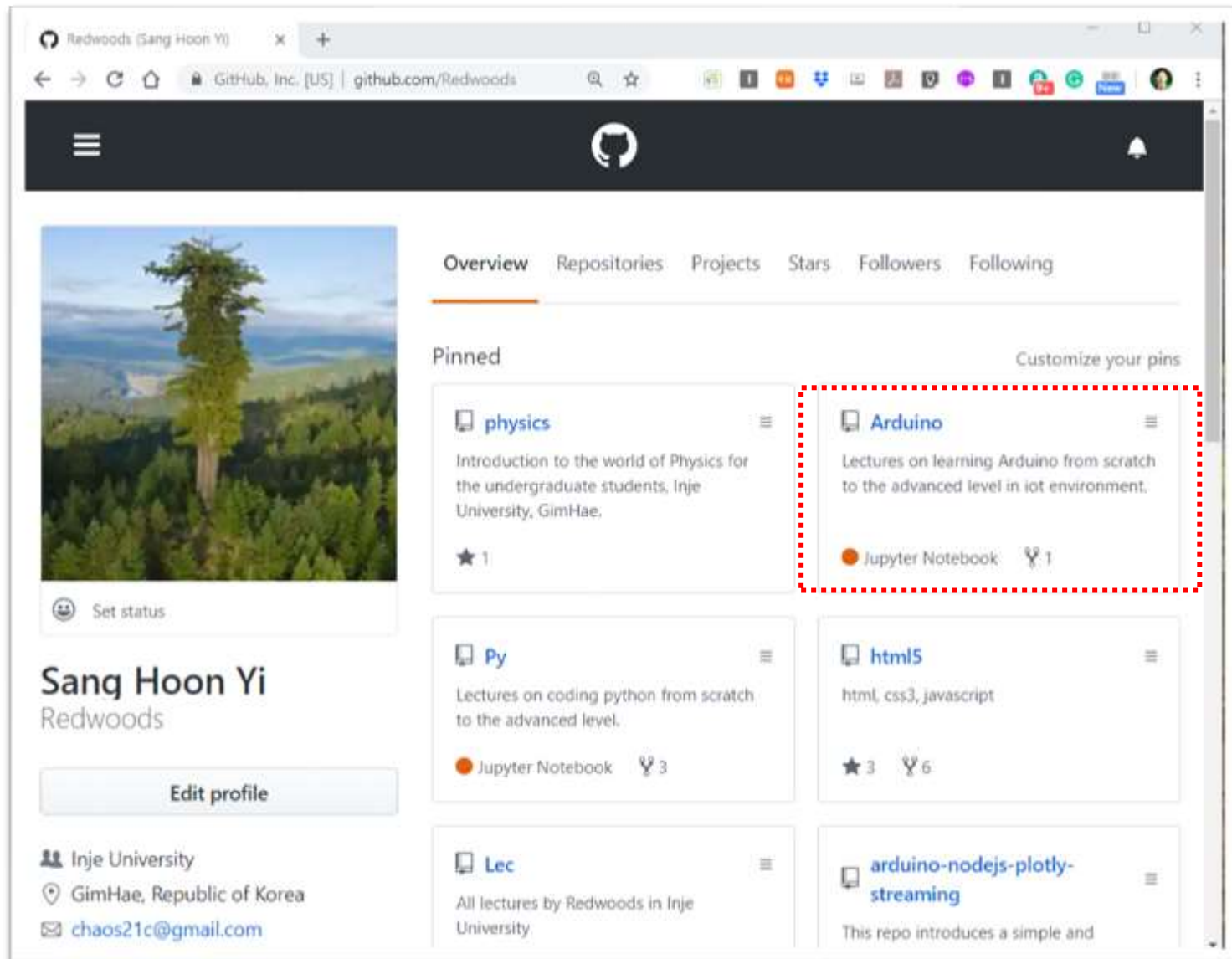
# TMP36 + CdS + LCD : circuit



```
1 // temperature & lux
2 #define TMP36_INPUT 0
3 #define CDS_INPUT 1
```

## ● 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

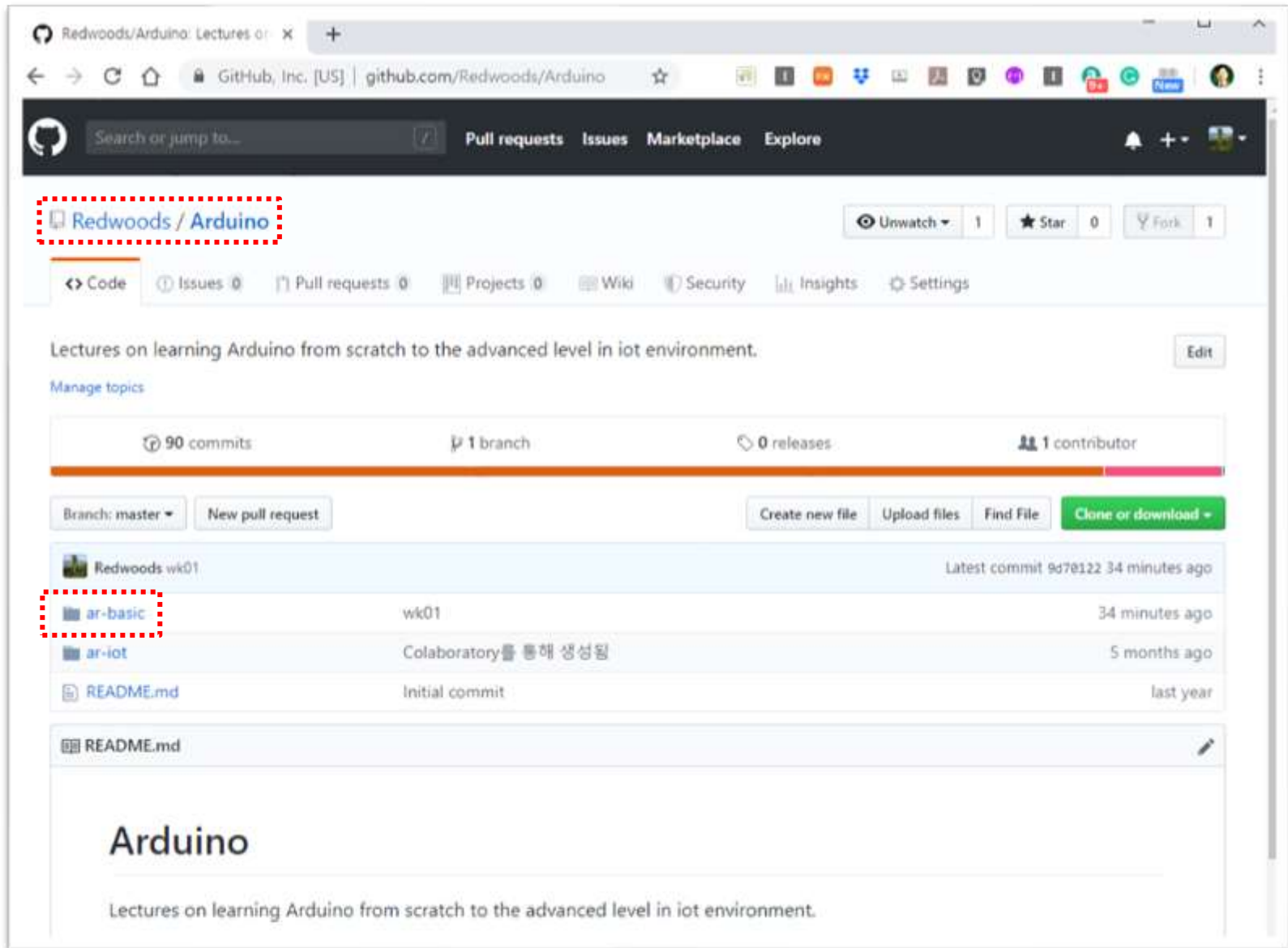
**html5**  
html, css3, javascript  
★ 3 6

**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 9d7e122	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

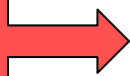
Lectures on learning Arduino from scratch to the advanced level in iot environment.



# 아두이노 키트(Kit)



**web**



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



## 아두이노 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	눌 스위치	리드 스위치 센서	4x4키 매트릭스	5V 릴레이 모듈	조이스틱 모듈	수위 센서
						
온도센서 LM35	저미스터	온습도센서	CDS 조도센서	불꽃감지센서	적외선 수신기	IR 리모콘

# 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	