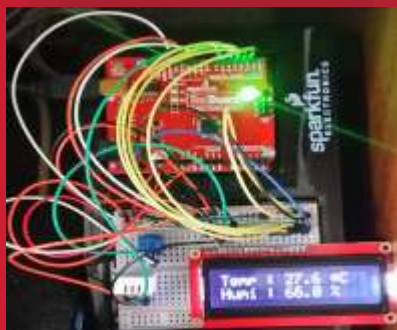




# Arduino-basic

## [wk13]

# Infrared remote



Learn how to code Arduino from scratch

Comsi, INJE University

2<sup>nd</sup> semester, 2018

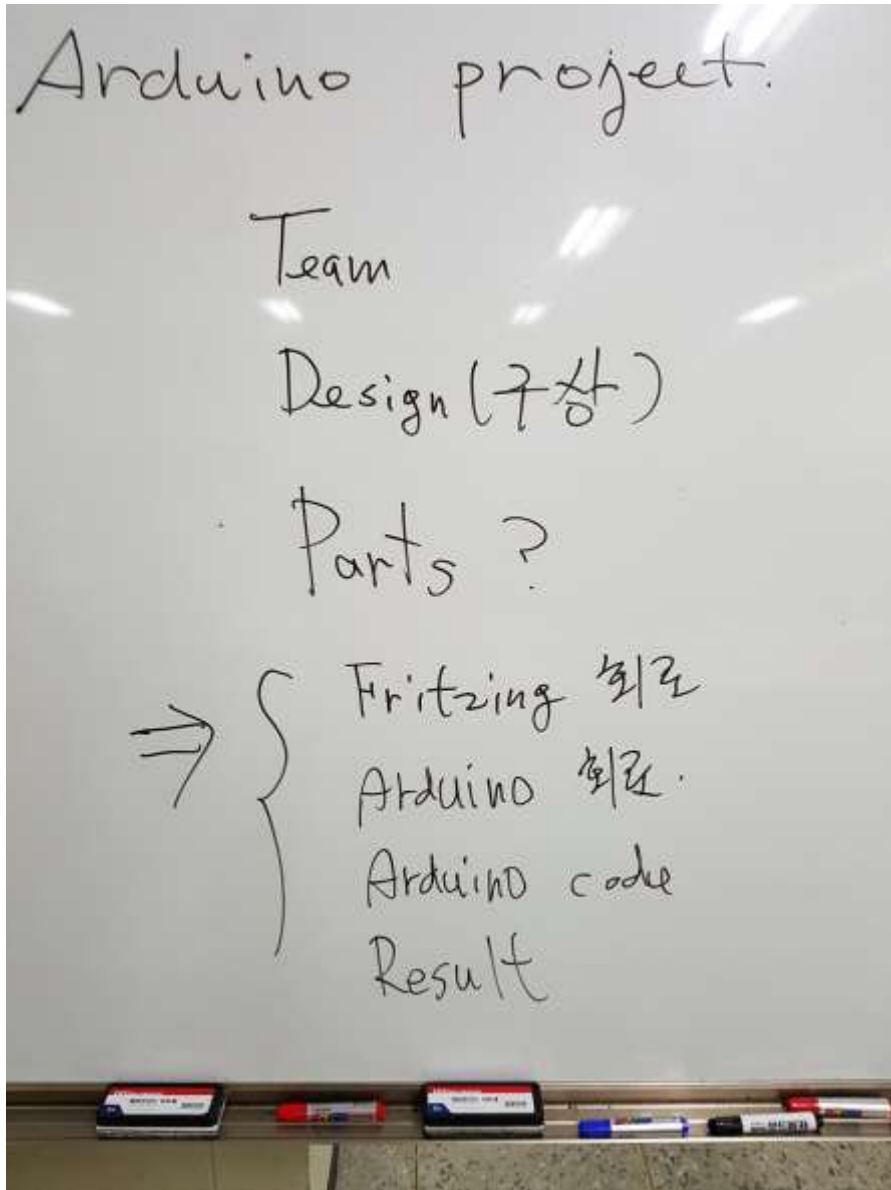
Email : [chaos21c@gmail.com](mailto:chaos21c@gmail.com)

# My ID (ARnn)

성명	ID
백동진	AR01
김도훈	AR02
김희찬	AR03
류재현	AR04
문민규	AR05
박진석	AR06
이승현	AR07
이승협	AR08
이후정	AR09
최민구	AR10

김다영	AR11
공진영	AR12
김해인	AR13
류성현	AR14
류재환	AR15
박상현	AR16
박해주	AR17
백지혜	AR18
송원식	AR19
신송주	AR20
윤지훈	AR21
정은성	AR22

# Arduino team project



- 2명/팀
- 구상 소개 (11.22, 11.29), ppt 준비
- 부품은 수업 세트 기준 (추가신청은 22일까지)
- 팀당 발표 자료 준비
- 발표 : 12월6일
- 참고



# [Review]

## ◆ [wk12]

- **Arduino : Motors**
- **Complete your project**
- **Submit file : ARnn\_Rpt09.zip**

# wk12 : Practice-09 : ARnn\_Rpt09.zip

## ◆ [Target of this week]

- Complete your works
- Save your outcomes and compress all.

제출파일명 : **ARnn\_Rpt09.zip**

- 압축할 파일들

- ① **ARnn\_sound\_bar.png**
- ② **ARnn\_sound\_monitor.png**
- ③ **ARnn\_step\_motor.png**
- ④ **ARnn\_step\_motor.ino**

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

**[ 제목 : id, 이름 (수정) ]**

# wk12 : Practice-09 : ARnn\_Rpt09.zip

```
const int inputPin1=2;
const int inputPin2=3;
const int inputPin3=4;
void setup() {
    //모터 신호핀을 출력으로 설정
    pinMode(motorPin1, OUTPUT);
    pinMode(motorPin2, OUTPUT);
    pinMode(motorPin3, OUTPUT);
    pinMode(motorPin4, OUTPUT);
    pinMode(inputPin1, INPUT_PULLUP);
    pinMode(inputPin2, INPUT_PULLUP);
    pinMode(inputPin3, INPUT_PULLUP);

    // 시리얼 통신 설정
    Serial.begin(9600);
}

void loop(){

    int potentiometer = analogRead(potentiometerPin);
    if(digitalRead(inputPin1)==LOW) {
        clockwise();
        Serial.println("cw");
    }

    if(digitalRead(inputPin2)==LOW) {
        counterClockwise();
        Serial.println("ccw");
    }

    if(digitalRead(inputPin3)==LOW) {
        motorStop();
        Serial.println("stop");
    }
}
```

```
const int cwPin = 2;
const int stopPin = 3;
const int ccwPin = 4;
boolean c = false;
boolean st_op = true;
boolean c_c = false;
void setup() {
    //모터 신호핀을 출력으로 설정
    pinMode(motorPin1, OUTPUT);
    pinMode(motorPin2, OUTPUT);
    pinMode(motorPin3, OUTPUT);
    pinMode(motorPin4, OUTPUT);
    pinMode(ccwPin, INPUT_PULLUP);
    pinMode(stopPin, INPUT_PULLUP);
    pinMode(cwPin, INPUT_PULLUP);
    // 시리얼 통신 설정
    Serial.begin(9600);
}

void loop(){

    int cw = digitalRead(cwPin);
    int st = digitalRead(stopPin);
    int ccw = digitalRead(ccwPin);
    if(cw == LOW){
        c = true;
        st_op = false;
        c_c = false;
    }
    else if(st == LOW){
        c = false;
        st_op = true;
        c_c = false;
    }
    else if(ccw == LOW){
        c = false;
        st_op = false;
        c_c = true;
    }
    // CW로 회전시 모터를 CW방향으로 회전시킨다.
    if(c == true){
        //모터의 속도를 계산한다.

        // 시리얼 통신 메시지를 출력한다.
        Serial.print("CW Motor Speed: ");
        // CW로 회전시킨다.
        clockwise();
    }
}
```

# wk12 : Practice-09 : ARnn\_Rpt09.zip



```
void loop() {
  int InCW = digitalRead(CW);
  int InCCW = digitalRead(CCW);
  int InSTOP = digitalRead(STOP);
  // 포텐쇼미터 값을 읽어옴
  if (InCW == LOW)
  {
    motorSpeed = map(1023, 512 + (stopRange / 2), 1023, 4500, 1000);
    motorSpeedPercent = map(motorSpeed, 4500, 1000, 1, 100);
    Serial.print("CW Motor Speed: ");
    Serial.print(motorSpeedPercent);
    Serial.println("%");
    // CW로 회전시킨다.
    clockwise();
  }
  else if (InCCW == LOW)
  {
    motorSpeed = map(1023, 512 - (stopRange / 2), 0, 4500, 1000);
    motorSpeedPercent = map(motorSpeed, 4500, 1000, 1, 100);
    Serial.print("CCW Motor Speed: ");
    Serial.print(motorSpeedPercent);
    Serial.println("%");
    // CCW로 회전시킨다.
    counterClockwise();
  }
  else
  {
    Serial.println("Motor Stop");
    motorStop();
  }
}
```

```
bool CWstate = false;
bool CCWstate = false;
void loop() {
  int potentiometer = analogRead(potentiometerPin);
  if(digitalRead(SW_W) == LOW || CWstate){
    motorSpeedPercent = 100;
    // 시리얼 통신 메시지를 출력한다.
    Serial.print("CW Motor Speed: ");
    Serial.print(motorSpeedPercent);
    Serial.println("%");
    CWstate = true;
    CCWstate = false;
    // CW로 회전시킨다.
    clockwise();
  }
  if(digitalRead(SW_B) == LOW || CCWstate){
    motorSpeedPercent = 100;
    // 시리얼 통신 메시지를 출력한다.
    Serial.print("CCW Motor Speed: ");
    Serial.print(motorSpeedPercent);
    CCWstate = true;
    CWstate = false;
    Serial.println("%");
    // CCW로 회전시킨다.
    counterClockwise();
  }
  if(digitalRead(SW_Y) == LOW){
    Serial.println("Motor Stop");
    motorStop();
    CCWstate = false;
    CWstate = false;
  }
}
```



# wk12 : Practice-09 : ARnn\_Rpt09.zip

```

bool cwstate =false;
bool ccwstate =false;

void loop(){
  // 포텐쇼미터 값을 읽어옴
  int potentiometer = analogRead(potentiometerPin);

  bool swVal1 = digitalRead(swPin1);
  bool swVal2 = digitalRead(swPin2);
  bool swVal3 = digitalRead(swPin3);

  // CW로 회전시 모터를 CW방향으로 회전시킨다.
  if((swVal1 == HIGH && swVal2 == HIGH && swVal3 == LOW) || cwstate){
    //모터의 속도를 계산한다.
    motorSpeedPercent = 99;
    Serial.print("CW Motor Speed: ");
    Serial.print(motorSpeedPercent);
    Serial.println("%");

    cwstate = true;
    ccwstate = false;
    // CW로 회전시킨다.
    clockwise();
  }
  // CCW로 회전시 모터를 CCW방향으로 회전시킨다.
  if((swVal1 == LOW && swVal2 == HIGH && swVal3 == HIGH) || ccwstate){
    //모터의 속도를 계산한다.

    motorSpeedPercent = 99;
    Serial.print("CCW Motor Speed: ");
    Serial.print(motorSpeedPercent);
    Serial.println("%");

    cwstate = false;
    ccwstate = true;

    // CCW로 회전시킨다.
    counterClockwise();
  }
  // 중간에 위치 했을 경우 정지시킨다.
  if(swVal2 == LOW){
    Serial.println("Motor Stop");

    cwstate = false;
    ccwstate = false;
    motorStop();
  }
}

```

```

// 모터 속도 관련 변수 설정
int motorSpeed=4000; //
int motorSpeedPercent;

```

```

void counterClockwise(){
  // 0~7 번째 신호를 순차적으로 출력시킨다.
  for(int i = 0; i < 8; i++){
    {
      motorSignalOutput(i);
      delayMicroseconds(motorSpeed);
    }
  }
}

```

```

void clockwise(){
  // 7~0 번째 신호를 순차적으로 출력시킨다.
  for(int i = 7; i >= 0; i--){
    {
      motorSignalOutput(i);
      delayMicroseconds(motorSpeed);
    }
  }
}

```

```

void motorStop(){
  // 정지신호를 출력시킨다.
  motorSignalOutput(8);
}

```





# 7. Motor driving



# 7. 모터 구동

7.1 스텝모터 구동

7.2 서보모터 구동

7.3 DC모터 구동

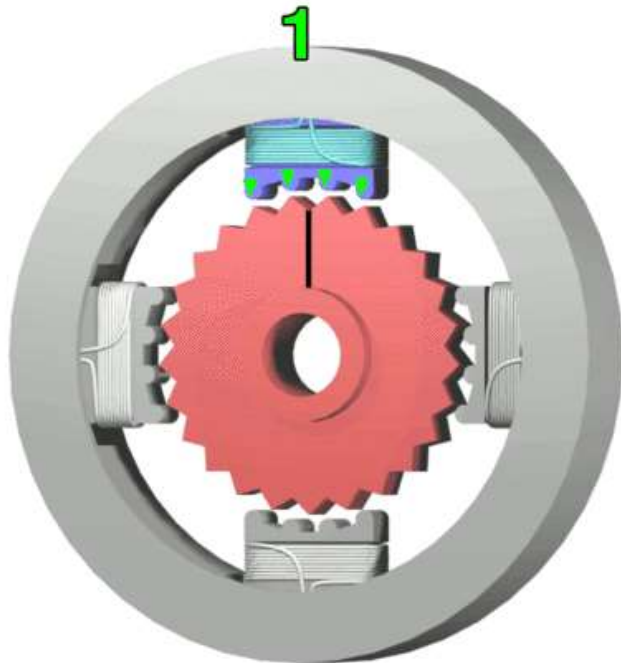


# 7.1

# 스텝모터



# 7.1 스텝모터: 구동 원리



## 스텝모터란

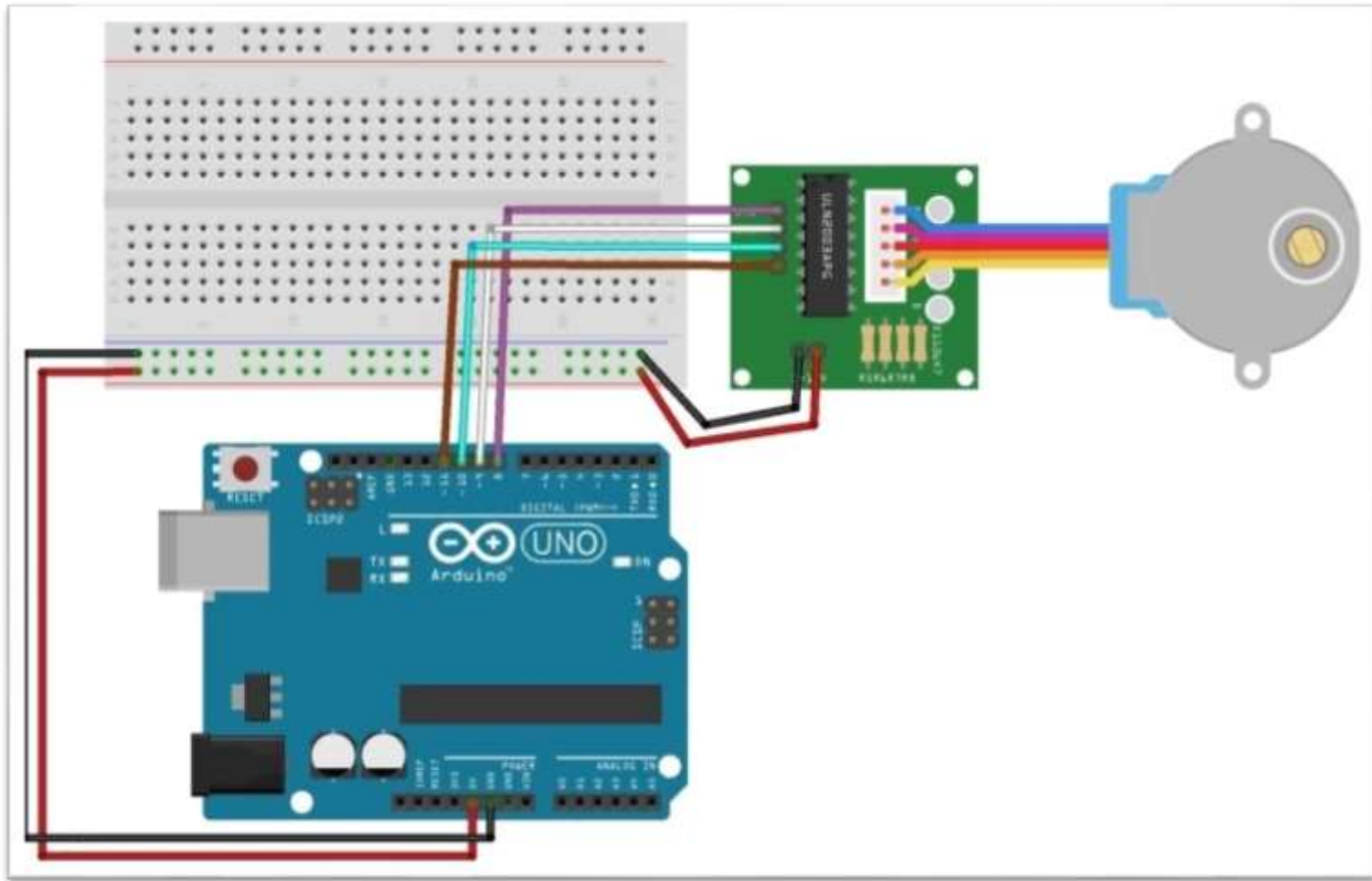
- 모터의 회전을 잘게 쪼개서 쪼개진 조각(스텝)을 이용해서 제어하는 모터.
- 스텝모터는 펄스(Puls)에 의해 디지털적으로 제어하기 때문에 정밀한 제어가 가능하고, 아두이노, 라즈베리파이등으로 동작시키기에 편리함.
- 제어부가 비교적 간단해서 가격이 저렴하고 브러시리스(Brushless)모터이기 때문에 오래 사용할 수 있다.

[https://en.wikipedia.org/wiki/Stepper\\_motor](https://en.wikipedia.org/wiki/Stepper_motor)

<https://m.blog.naver.com/3demp/220976664782>

EX 7.1

스텝모터 구동 (1/3)



<https://m.blog.naver.com/3demp/220976664782>

## 7.1.3 스텝모터 : code-3

```

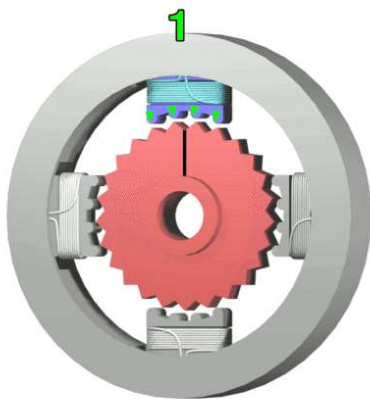
74 void counterClockwise(){
75     // 0~7 번째 신호를 순차적으로 출력시킨다.
76     for(int i = 0; i < 8; i++)
77     {
78         motorSignalOutput(i);
79         delayMicroseconds(motorSpeed);
80     }
81 }
82
83 void clockwise(){
84     // 7~0 번째 신호를 순차적으로 출력시킨다.
85     for(int i = 7; i >= 0; i--)
86     {
87         motorSignalOutput(i);
88         delayMicroseconds(motorSpeed);
89     }
90 }

```

```

92 void motorStop(){
93     // 정지신호를 출력시킨다.
94     motorSignalOutput(8);
95 }
96
97 void motorSignalOutput(int out)
98 {
99     // out 변수에 해당하는 모터 시그널을 출력한다.
100     digitalWrite(motorPin1, bitRead(steps[out], 0));
101     digitalWrite(motorPin2, bitRead(steps[out], 1));
102     digitalWrite(motorPin3, bitRead(steps[out], 2));
103     digitalWrite(motorPin4, bitRead(steps[out], 3));
104 }

```



bitRead(x, n)

매개변수

x: 읽을 숫자

n: 읽을 비트, LSB(맨 오른쪽 비트)가 0, 왼쪽으로 갈수록 1씩 증가

// 스텝 모터의 스텝 설정

// 0~7은 동작 신호, 8번째는 모터 정지 신호

```
int steps[] = {B1000, B1100, B0100, B0110, B0010, B0011,
B0001, B1001, B0000};
```



## 7.1.5 스텝모터 : DIY

### DIY

### 응용 문제

1. 3개의 스위치를 디지털 입력으로 연결하자.
2. CW, STOP, CCW 기능을 하도록 스케치를 작성하자.
3. STOP일 경우 모든 신호는 스텝모터로 입력되지 않아야 한다.

아두이노 회로가 포함된 동작 사진을  
**ARnn\_step\_motor.png**

아두이노 스케치 코드를  
**ARnn\_step\_motor.ino** 로  
저장...



# 7.2

## 서보모터



## 서보모터 (Servo Motor)

- ✓ 기계적인 위치, 속도, 가속도 등을 제어하는 모터
- ✓ 산업용 서보모터는 로봇의 관절, 공작 기계의 위치제어 등에 사용
- ✓ RC용 서보모터는 RC 자동차나 RC 비행기에 사용



그림 7. 4 실험에 사용할 소형 RC 서보모터 (SG90)



그림 7. 3 산업용 서보모터(a)와 RC용 서보모터(b)

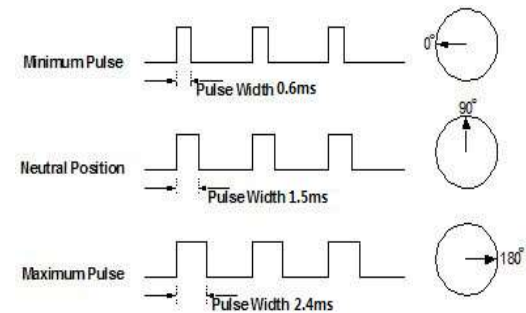
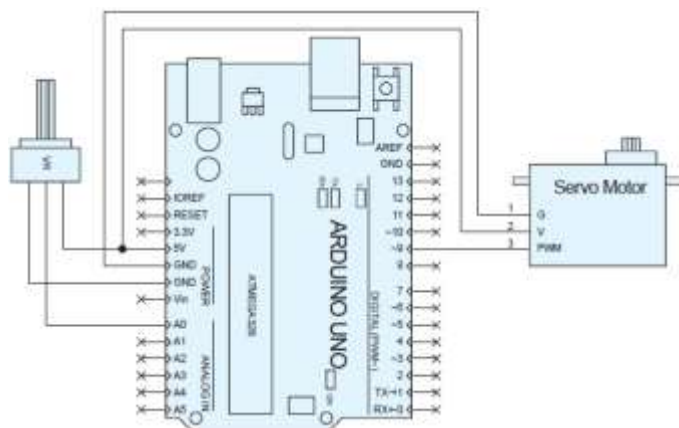


그림 7. 5 PWM 신호와 RC 서보모터의 회전각

## EX 7.2 서보모터 구동 (1/3)

- 실습목표**
1. 소형 RC용 서보모터를 구동한다.
  2. 포텐쇼미터의 각도에 따라서 서보모터의 각도를 조절한다.
  3. 현재 각도를 시리얼 통신으로 전송한다.

- Hardware**
1. 포텐쇼미터의 1, 3번핀을 Arduino의 5V, GND에 연결한다.
  2. 포텐쇼미터의 2번핀을 Arduino의 아날로그입력핀 A0에 연결한다.
  3. 서보모터의 V(적색)와 GND(검정 혹은 갈색)핀을 Arduino의 5V와 GND에 연결한다.
  4. Arduino의 9번핀을 서보모터의 **PWM**핀(흰색 혹은 주황)과 연결한다.



## EX 7.2

## 서보모터 구동 (2/3)

**Commands** • `Serial.begin(전송속도)`

시리얼 통신 포트를 컴퓨터와 연결한다. 전송속도는 bps(bits per sec)로 일반적으로 9600으로 설정한다. 19200, 57600, 115200 등의 값을 설정할 수 있다.

• `Serial.print(전송내용)`

괄호 안의 내용을 시리얼 통신으로 전송한다. 따옴표로 구분된 부분은 텍스트를 직접 전송하고 따옴표 없이 변수를 써주면 변수의 값이 전송된다.

• `Serial.println(전송내용)` 'Serial.print'와 같으나 전송 뒤 줄 바꿈을 한다.

• `서보모터이름.attach(핀번호)`

이름을 설정한 서보모터를 핀번호에 설정한다.

• `서보모터이름.attach(핀번호, 최소펄스, 최대펄스)`

이름을 설정한 서보모터를 핀번호에 설정한다. 서보모터가 동작하는 최소 펄스와 최대 펄스를 마이크로세컨드 단위로 설정한다.

• `서보모터이름.write(각도)`

이름을 설정한 서보모터를 정해진 각도로 위치시킨다.

## 7.2.3 서보모터

### EX 7.2 서보모터 구동 (3/3)

- Sketch 구성**
1. 서보모터 라이브러리를 추가한다.
  2. 서보모터 이름을 설정하고 9번핀을 서보모터 출력으로 사용한다.
  3. 서보모터의 회전범위를 설정한다 라이브러리의 `attach()` 함수를 사용한다.
  4. 포텐쇼미터값을 아날로그 0번핀으로 입력받아 0~1023 범위의 포텐쇼미터값 범위를 0~180도로 환산하여 서보모터를 동작시킨다.

- 실습 결과**
1. 포텐쇼미터의 회전각에 따라 서보모터가 회전한다.
  2. 포텐쇼미터 값이 변화하면 시리얼 통신으로 회전각을 전송한다.
  3. 현재의 각도를 LCD 모듈로 출력한다.



## 7.2.4 서보모터: code

ex\_7\_2

```

1 /*
2  예제 7.2
3  서보모터 구동
4 */
5
6 // 서보모터 라이브러리 불러오기
7 #include <Servo.h>
8
9 // 서보모터 이름 설정
10 Servo motor1;
11
12 // 서보 모터 신호핀 설정
13 int servoMotorPin = 9;
14
15 // 포텐쇼미터 핀 설정
16 int potentiometerPin = 0;
17
18 // 모터 각도 변수 설정
19 int motorAngle;
20 int motorAngleOld;
21

```

```

22 void setup() {
23
24   // 서보모터 설정. 0.6ms 부터 2.4ms 범위로 설정
25   motor1.attach(servoMotorPin, 600, 2400);
26
27   // 시리얼 통신 설정
28   Serial.begin(9600);
29 }
30
31 void loop(){
32   // 포텐쇼미터 값을 읽어옴
33   int potentiometer = analogRead(potentiometerPin);
34
35   // 포텐쇼미터 값을 모터 각도로 변환한다
36   motorAngle = map(potentiometer, 0, 1023, 0, 180);
37
38   // 모터에 각도값을 전달한다
39   motor1.write(motorAngle);
40
41   // 이전각도와 현재 각도가 같지 않으면 시리얼 모니터에 각도를 출력한다.
42   if(motorAngle != motorAngleOld){
43     Serial.print("Servo Motor Angle is: ");
44     Serial.println(motorAngle);
45   }
46
47   // 현재의 모터 각도를 저장한다.
48   motorAngleOld = motorAngle;
49
50   delay(20);
51
52 }

```

## 7.2.5 서보모터 - DIY

DIY

1. 좌, 우 두 개의 스위치 입력을 받는다.

응용 문제

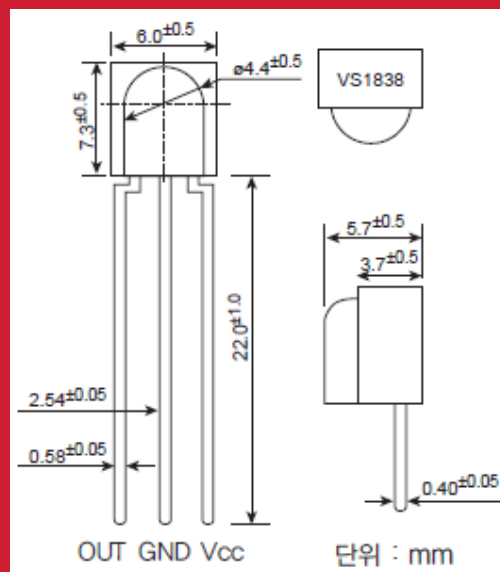
2. 스위치를 누를 때 마다 해당 방향으로 회전하게 한다.

아두이노 스케치 코드를

**ARnn\_servo motor.ino** 로  
저장...



# 8. Infrared remote



## 8. Infrared remote

**8.1 적외선 리모컨 코드 읽기**

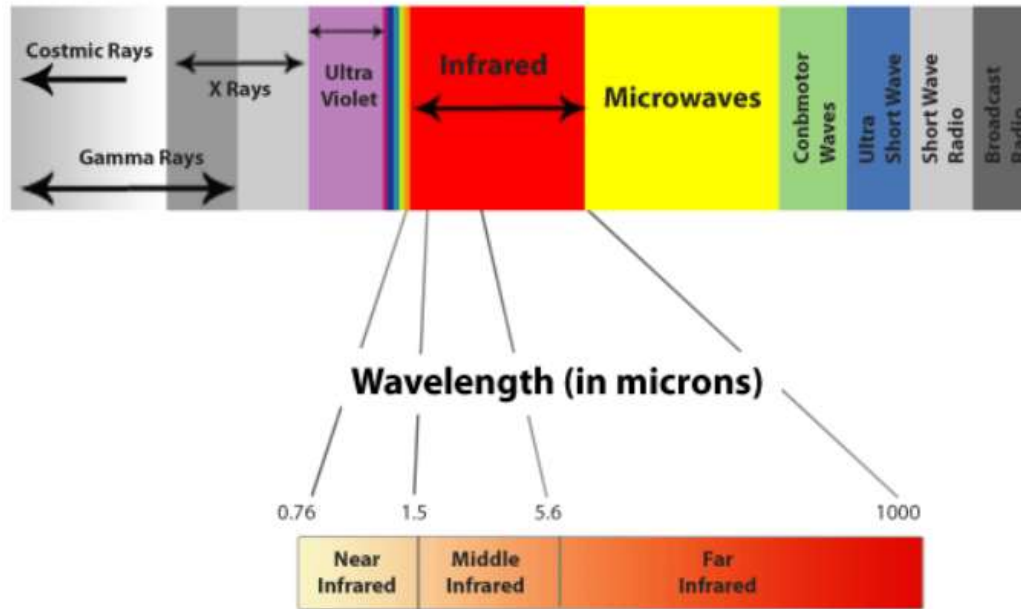
**8.2 적외선 리모컨을 이용한 LED 제어**



# 8.1

## 적외선 리모컨





전자기파

<적외선의 파장은 0.75um부터 1,000um사이에 위치합니다>



<https://kocoafab.cc/tutorial/view/703>



## 적외선 리모컨



그림 8.1 실험에 사용할 리모컨과 수신부

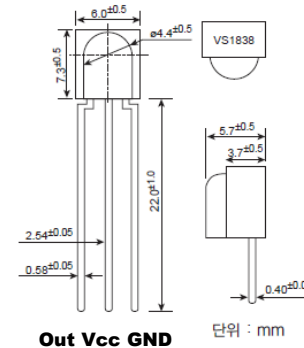


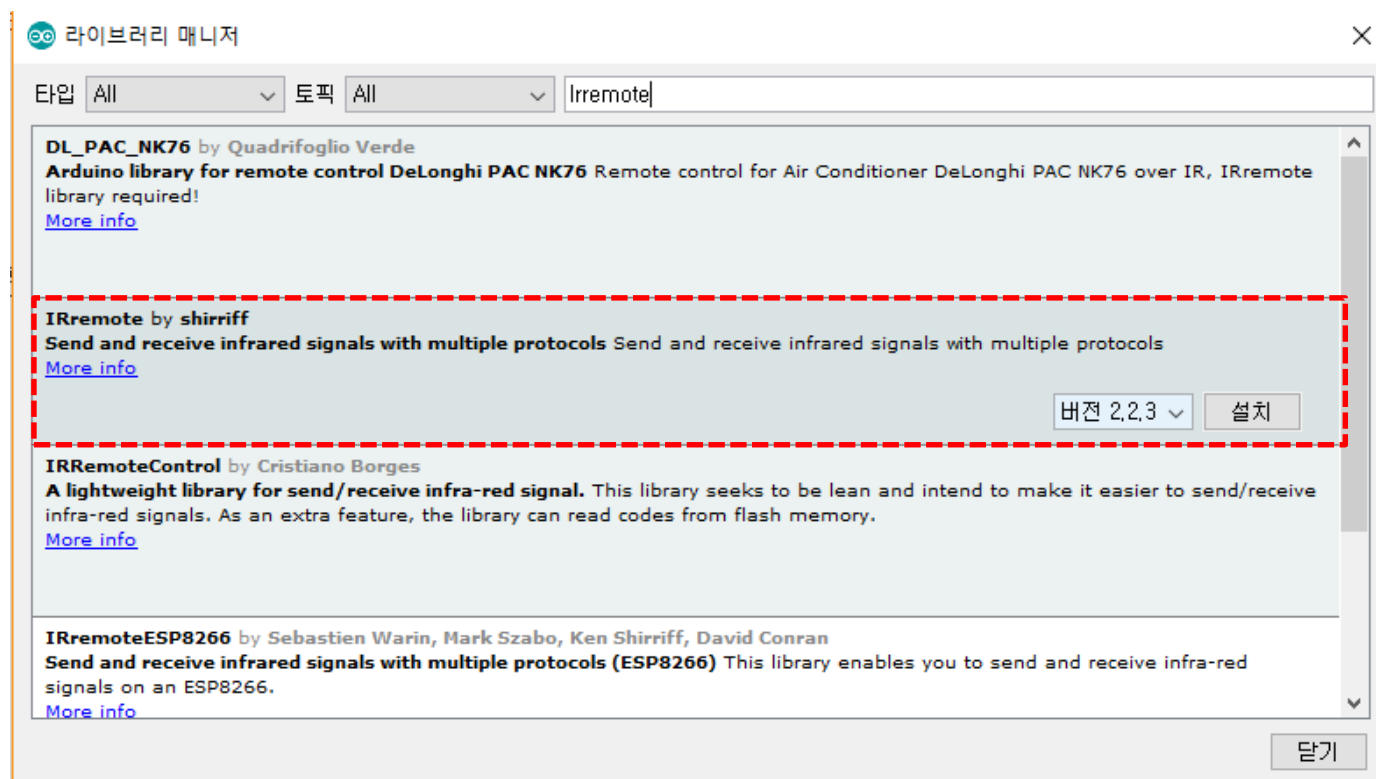
그림 8.2 리모컨 수신부의 핀 설정

- ✓ TV, 에어컨 등 대부분의 가전제품에 사용
- ✓ 적외선 램프의 점멸을 이용하여 데이터 송 수신
- ✓ 송신 시 캐리어 주파수로 변조하여 전송. 수신부와 캐리어주파수가 일치하여야함

※ 실험에 사용할 라이브러리를 하기의 주소에서 다운받아 설치할 것

<https://github.com/shirriff/Arduino-IRremote>

## 적외선 리모컨 - 라이브러리 설치



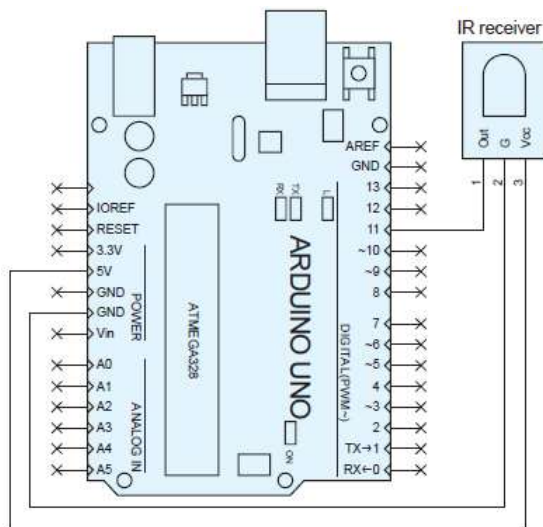
✓ 라이브러리 매니저를 실행 한 후 IRremocon으로 검색하여 라이브러리를 설치

# 8.1.1 적외선 리모컨

## EX 8.1 적외선 리모컨 코드 읽기 (1/3)

- 실습목표**
1. 적외선 리모컨의 신호를 수신한다.
  2. 신호가 수신 될 때 내장된 13번 핀 LED를 점멸시킨다.
  3. 수신된 코드를 시리얼 통신으로 전송한다.
  4. 리모컨 버튼마다 어떤 코드가 수신되는지 확인한다.

- Hardware**
1. IR Receiver의 Vcc와 G를 Arduino의 5V와 GND에 연결한다.
  2. IR Receiver Out은 디지털입출력 11번 핀에 연결한다.



## 8.1.2 적외선 리모컨

### EX 8.1

### 적외선 리모컨 코드 읽기 (2/3)

#### Commands

- **IRrecv 리모컨이름(리모컨 수신 핀번호)**  
리모컨 이름으로 리모컨 수신핀을 설정한다.
- **리모컨이름.enableIRIn()**  
리모컨 수신시 타이머 인터럽트를 활성화한다.
- **decode\_result 실습결과**  
입력된 신호의 결과를 '실습결과' 변수에 입력한다.
- **리모컨이름.decode(&실습결과)**  
리모컨 수신이 이루어졌을 때 그 값을 회신한다.
- **실습결과.decode\_type**  
수신된 리모컨 코드의 타입이 저장된다(NEC, SONY, RC5, RC6 등).
- **실습결과.value**  
수신된 리모컨의 코드가 저장된다.
- **실습결과.bits**  
수신된 리모컨 코드의 비트수가 저장된다.
- **실습결과.rawbuf**  
수신된 적외선 펄스의 시간이 배열형태로 저장된다.
- **실습결과.rawlen**  
수신된 적외선 펄스의 시간이 배열의 개수가 저장된다.
- **리모컨이름.resume()**  
수신 후에 다른 신호를 수신하기 위하여 준비한다.
- **리모컨이름.blink13(true)**  
Arduino에 내장된 LED(13번핀)을 리모컨 수신할 때 마다 점멸시킨다.

## 8.1.3 적외선 리모컨

### EX 8.1 적외선 리모컨 코드 읽기 (3/3)

- Sketch 구성**
1. 리모컨 라이브러리를 불러온다.
  2. 디지털입력 11번 핀을 리모컨 수신핀으로 설정한다.
  3. 'irrecv'라는 이름으로 리모컨이름을 설정한다.
  4. 리모컨신호 수신 결과를 'results'라는 이름으로 설정한다.
  5. 리모컨 수신이 발생했을 때 수신 값을 시리얼 통신으로 전송한다.

- 실행 결과**
1. 'Received Code is 수신된 코드' 의 메시지가 출력된다.
  2. 키를 누를 때 마다 수신된 코드가 변경된다.
  3. 모든 키에 대하여 수신된 코드를 표로 완성하자.

## 8.1.4 적외선 리모컨: code

ex\_8\_1

```

1 /*
2  예제 8.1
3  적외선 리모컨 코드 읽기
4 */
5
6 // 적외선 리모컨 라이브러리를 불러온다.
7 #include <IRremote.h>
8
9 // 적외선 수신부가 연결될 핀을 설정한다.
10 int irPin = 11;
11
12 // 적외선 수신부가 연결된 핀을 리모컨 수신 핀으로 설정한다.
13 IRrecv irrecv(irPin);
14
15 // 수신된 신호의 결과를 results 변수로 설정한다.
16 decode_results results;
17

```

```

18 void setup()
19 {
20   // 시리얼 통신을 설정한다.
21   Serial.begin(9600);
22
23   // 적외선 리모컨 수신을 시작한다.
24   irrecv.enableIRIn();
25
26   // 13번 핀에 연결된 LED를 리모컨 수신시 점멸시킨다.
27   irrecv.blink13(true);
28
29 }
30
31 void loop()
32 {
33   // 수신된 코드가 있을 때 실행한다.
34   if (irrecv.decode(&results)){
35     // 0xFFFFFFFF 값을 제외하고 출력한다.
36     if(results.value != 0xFFFFFFFF){
37       // 수신된 값을 16진수 형태로 출력한다.
38       Serial.print("Received Code is ");
39       Serial.println(results.value, HEX);
40     };
41     // 다음 수신을 위해서 준비한다.
42     irrecv.resume();
43   }
44 }

```



## 8.1.5 적외선 리모컨: DIY

DIY

응용 문제

1. LCD 모듈에 수신된 코드를 표시해 보자.
2. 집에서 사용하는 TV의 리모컨의 코드를 확인해 보자.

수신된 코드가 출력된 **LCD** 사진을  
**ARnn\_remote LCD.png** 로  
 저장...

COM11 (Arduino/Genuino Uno)

```
Received Code is 12B07291
Received Code is 293C1067
Received Code is 4AB0F7B5
Received Code is FFA857
Received Code is FFA857
Received Code is FFA857
Received Code is FFA857
Received Code is FFA857
Received Code is FFA857
Received Code is FF38C7
Received Code is 488F3CBB
```



## 8.2

# 적외선 리모컨으로 LED 제어

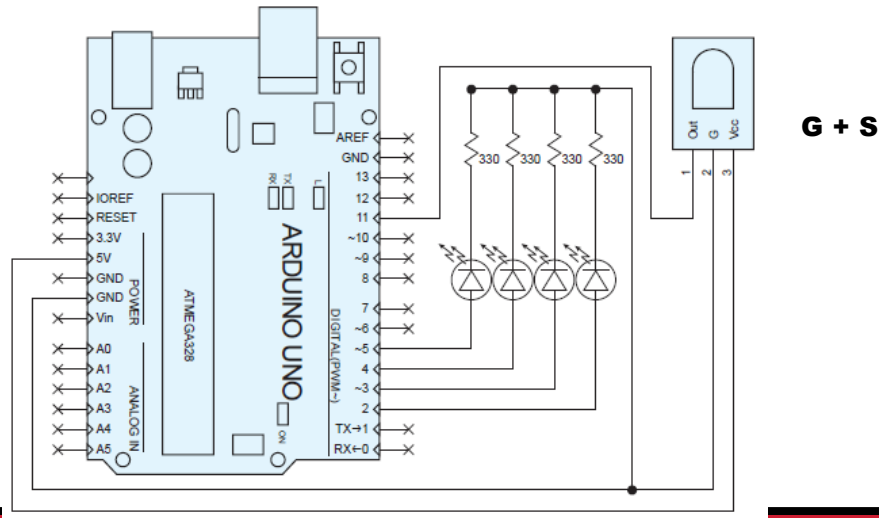


# 8.2.1 적외선 리모컨을 이용한 LED 제어

## EX 8.2 적외선 리모컨을 이용한 LED 제어 (1/2)

- 실습목표**
1. 적외선 리모컨의 신호를 수신한다.
  2. 수신된 코드에 따라 LED를 제어한다.
  3. 메시지를 시리얼 통신으로 전송한다.

- Hardware**
1. IR Receiver의 Vcc와 G를 Arduino의 5V와 GND에 연결한다.
  2. IR Receiver Out은 디지털입출력 11번 핀에 연결한다.
  3. Arduino의 디지털 입출력 2~5번핀에 4개의 LED의 Anode 핀을 연결한다.
  4. 각 LED의 Cathode에는 330Ω 저항을 연결하여 GND에 연결한다.



## 8.2.2 적외선 리모컨을 이용한 LED 제어

### EX 8.2 적외선 리모컨을 이용한 LED 제어 (3/3)

- Sketch 구성**
1. 리모컨 라이브러리를 불러온다.
  2. 디지털입력 11번 핀을 리모컨 수신핀으로 설정한다.
  3. 'irrecv'라는 이름으로 리모컨이름을 설정한다.
  4. 리모컨신호 수신 결과를 'results'라는 이름으로 설정한다.
  5. 디지털 입출력핀 2, 3, 4, 5를 LED 출력으로 사용하기 위해 출력모드로 설정한다.
  6. 각 LED의 점등과 소등 신호를 미리 설정해 준다.
  7. 적외선 리모컨 신호가 수신되었을 때 신호에 맞춰 LED를 동작시킨다.

- 실습 결과**
1. 리모컨 키를 누를 때 마다 해당 LED가 점등/소등 한다.
  2. 현재 동작 상태에 대한 메시지가 시리얼 통신으로 전송된다.

## 8.2.3 적외선 리모컨을 이용한 LED 제어:code

```

1 /*
2  * 예제 8.2
3  * 적외선 리모컨을 이용한 LED 제어
4  */
5
6 // 적외선 리모컨 라이브러리를 불러온다.
7 #include <IRremote.h>
8
9 // 적외선 수신부가 연결될 핀을 설정한다.
10 int irPin = 11;
11
12 // LED에 연결된 핀을 설정한다.
13 int led1 = 2;
14 int led2 = 3;
15 int led3 = 4;
16 int led4 = 5;
17
18 // LED 제어용 코드 (리모컨에 맞게 수정한다.)
19 long on1 = 0xFFA25D;
20 long off1 = 0xFF629D;
21 long on2 = 0x6182021B;
22 long off2 = 0x8C22657B;
23 long on3 = 0x488F3CB8;
24 long off3 = 0x449E79F;
25 long on4 = 0x3206FDF7;
26 long off4 = 0x1BC0157B;
27
28 // 적외선 수신부가 연결된 핀을 리모컨 수신 핀으로 설정한다.
29 IRrecv irrecv(irPin);
30
31 // 수신된 신호의 결과를 results 변수로 설정한다.
32 decode_results results;
33

```

```

34 void setup()
35 {
36   // 시리얼 통신을 설정한다.
37   Serial.begin(9600);
38
39   // 적외선 리모컨 수신을 시작한다.
40   irrecv.enableIRIn();
41
42   // 13번 핀에 연결된 LED를 리모컨 수신시 점멸시킨다.
43   irrecv.blink13(true);
44
45   pinMode(led1, OUTPUT);
46   pinMode(led2, OUTPUT);
47   pinMode(led3, OUTPUT);
48   pinMode(led4, OUTPUT);
49 }
50
51 void loop()
52 {
53   // 수신된 코드가 있을 때 실행한다.
54   if (irrecv.decode(&results)){
55     // 0xFFFFFFFF 값을 제외하고 출력한다.
56     if(results.value != 0xFFFFFFFF){
57       // 수신된 코드가 on1과 같을 때
58       if(results.value == on1){
59         digitalWrite(led1, HIGH);
60         Serial.println("LED1 is ON");
61       }
62       // 수신된 코드가 off1과 같을 때
63       if(results.value == off1){
64         digitalWrite(led1, LOW);
65         Serial.println("LED1 is OFF");
66       }
67     }
68   }
69 }

```

## 8.2.4 적외선 리모컨을 이용한 LED 제어

DIY

1. LCD 모듈에 수신된 코드를 표시해 보자.

응용 문제

2. 집에서 사용하는 TV의 리모컨을 사용하여 만들어 LED를 제어해 보자.

```
COM11 (Arduino/Genuino Uno)

LED1 is ON
LED1 is OFF
LED1 is OFF
LED1 is ON
LED1 is OFF
LED1 is ON
LED1 is OFF
LED1 is ON
LED1 is OFF
LED1 is ON
LED1 is OFF
LED1 is ON
LED1 is OFF
LED1 is ON
LED1 is OFF
```

수신된 코드가 출력된 LCD와 지정된  
LED가 켜진 사진을  
**ARnn\_remote\_LED.png** 로  
저장...

그리고 아두이노 스케치 코드를  
**ARnn\_remote\_LED.ino** 로 저장



# [Practice]

## ◆ [wk13]

- **Arduino : Infrared remote**
- **Complete your project**
- **Submit file : ARnn\_Rpt10.zip**

# wk13 : Practice-10 : ARnn\_Rpt10.zip

## ◆ [Target of this week]

- Complete your works
- Save your outcomes and compress all.

제출파일명 : **ARnn\_Rpt10.zip**

- 압축할 파일들

① **ARnn\_servo\_motor.ino**

② **ARnn\_remote\_LCD.png**

③ **ARnn\_remote\_LED.png**

④ **ARnn\_remote\_LED.ino**

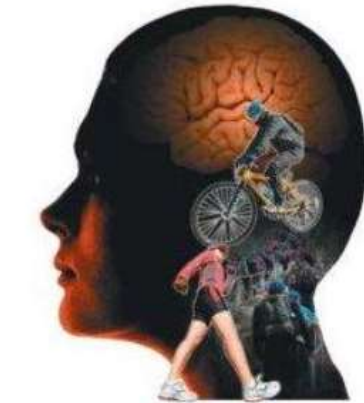
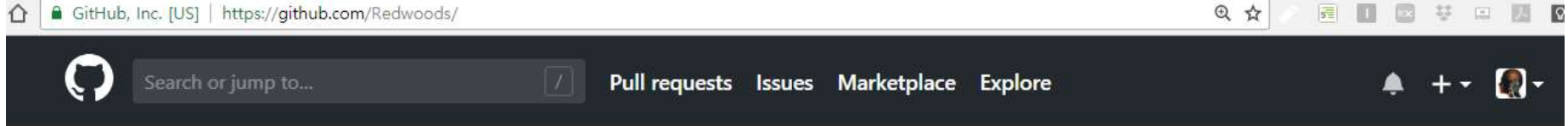
**Email : [chaos21c@gmail.com](mailto:chaos21c@gmail.com)**

**[ 제목 : id, 이름 (수정) ]**



## ● References & good sites

- ✓ <http://www.arduino.cc> Arduino Homepage
- ✓ <http://www.github.com> GitHub
- ✓ <http://www.google.com> Googling
- ✓ <https://www.youtube.com> Youtube



**Redwoods Yi**

Redwoods

Add a bio

GimHae, Republic of Korea

chaos21c@gmail.com

Overview

Repositories 7

Stars 2

Followers 1

Following 0

## Pinned repositories

Customize your pinned repositories

Py

Lectures on coding python from scratch to the advanced level.

Jupyter Notebook

Arduino

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

Lec

All lectures by Redwoods in Inje University

Jupyter Notebook

hw-coding

Resource for lecture of Hardware Programming (2017, Inje university)


Arduino

171 contributions in the last year




Contribution settings

Redwoods/Arduino: Lect

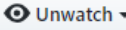
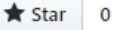

GitHub, Inc. [US] | https://github.com/Redwoods/Arduino

 Search or jump to...


[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)

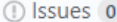
  

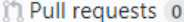
Redwoods / Arduino


 1  0  0


<> Code

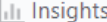
 Issues 0

 Pull requests 0

 Projects 0

 Wiki

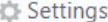
 Insights

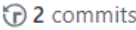
 Settings


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


Edit

Add topics

 2 commits

 1 branch

 0 releases

 1 contributor

Branch: master ▾

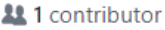
New pull request

Create new file


Upload files

Find file

Clone or download ▾


 Redwoods 2018 start

Latest commit 38ca9e0 28 minutes ago

 ar-basic


2018 start

28 minutes ago

 ar-iot


2018 start


28 minutes ago

 README.md

Initial commit

43 minutes ago

 README.md



## Arduino

---

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





[http://arduinostory.com/goods/goods\\_view.php?goodsNo=1000000306](http://arduinostory.com/goods/goods_view.php?goodsNo=1000000306)



## 상급키트 구성품

<b>1</b> 1EA  <b>아두이노 우노 R3 DIP</b> 아두이노 우노 R3 (DIP) 호환보드 기본 메인보드입니다.	<b>2</b> 1EA  <b>9V 배터리 홀더</b> 9V 배터리를 연결하여 아두이노에 외부전원을 공급할 수 있습니다.	<b>3</b> 1EA  <b>7세그먼트 4채널</b> 7세그먼트가 4개 연결된 형태의 부품입니다. 총 12개의 핀을 사용합니다.	<b>4</b> 1EA  <b>7세그먼트 1채널</b> 공통 음극 7세그먼트 시계나 침수 등의 숫자를 표현 할 때 많이 사용됩니다.	<b>13</b> 1EA  <b>수동부저</b> 아두이노의 tone함수를 통해 소리를 내는 부저입니다.	<b>14</b> 6EA  <b>택트스위치 (12x12x7)</b> 스위치를 누르고 있을 경우만 ON됩니다.	<b>15</b> 각3EA  <b>택트스위치 캡 (피랑,노랑,초록,빨강,하양)</b> 택트스위치를 사용할 때 스위치간의 구분을 할 수 있습니다.	<b>16</b> 3EA  <b>조도센서</b> 빛을 감지하거나 빛의 밝기를 아날로그로 출력해주는 CDS 센서입니다.
<b>5</b> 1EA  <b>74HC595N</b> 기본 메인보드입니다. 74HC595N LED, 드레드텍스, NFD 제어 IC 입니다.	<b>6</b> 1EA  <b>65핀 점퍼 와이어</b> 브레드보드에 연결할 때 사용하는 65핀 점퍼와이어 입니다.	<b>7</b> 1EA  <b>무지개 점퍼선 F-M 20cm</b> M타입과 F타입이 양쪽으로 달린 무지개 점퍼선입니다.	<b>8</b> 1EA  <b>투명 부품 케이스 대,소</b> 키트 구성품을 담을 수 있는 투명 부품 케이스입니다.	<b>17</b> 각5EA  <b>LED 5mm (빨강,노랑,초록,하양,파랑)</b> 기본으로 사용되는 LED입니다. 동작전압 : 2.2~2.4V 사용전류 : 20mA 미만	<b>18</b> 1EA  <b>헤더핀 1x40/2.54mm</b> 핀 간격은 2.54mm이며 헤더핀의 길이는 약 1.15cm입니다.	<b>19</b> 1EA  <b>USB케이블 50cm</b> PC와 아두이노 우노 보드를 연결하여 프로그램을 다운로드 할 때 사용합니다.	<b>20</b> 1EA  <b>저항값 카드</b> 저항값을 쉽게 확인 할 수 있는 카드입니다. 사이즈 : 60mm x 50mm
<b>9</b> 1EA  <b>가변저항10K</b> 물리변 저항값이 가능합니다. (0~10KΩ)	<b>10</b> 1EA  <b>1602 I2C LCD</b> 아두이노 16x2 I2C LCD 모듈입니다. LCD입니다.	<b>11</b> 각 10EA  <b>저항</b> 100, 220, 330, 1K, 2K, 4.7K, 10K, 47K, 100K	<b>12</b> 1EA  <b>브레드 보드 830울</b> 브레드 보드 830울(봉무형) 센서 테스트나, 회로 프로토타입을 작성할 때 사용됩니다.	<b>21</b> 1EA  <b>능동부저</b> Signal 단자가 HIGH 일 때 약 2.5kHz의 음이 발생됩니다.	<b>22</b> 1EA  <b>5V 1채널 릴레이 모듈</b> 아두이노의 디지털 핀과 모듈 하단의 IN 핀들을 연결해 릴레이를 제어할 수 있는 모듈입니다.	<b>23</b> 1EA  <b>8x8 도트 매트릭스 모듈</b> LED로 다양한 연출을 할 수 있습니다.	<b>24</b> 1EA  <b>4x4 16 키패드 모듈</b> 16개의 버튼을 사용할 수 있습니다.

# 아두이노 키트(Kit) : Part-2

<p>25 1EA</p> <p>무선 리모콘 키트</p> <p>핵파선을 사용해서 리모콘 기능을 구현할 수 있습니다.</p>	<p>26 2EA</p> <p>가열기 센서 스위치</p> <p>센서의 가열기에 따라 스위치 역할을 합니다.</p>	<p>27 1EA</p> <p>or</p> <p>사운드 센서 모듈</p> <p>아두이노와 호환되는 사운드센서 모듈입니다.</p>	<p>28 1EA</p> <p>불꽃 센서</p> <p>근거리 화재, 불꽃을 감지하는 센서입니다.</p>	<p>37 1EA</p> <p>DC 5V 스텝 모터</p> <p>28BYJ-48 스텝 모터 중 저렴한 편에 속하는 모델입니다. 5개의 핀을 사용합니다.</p>	<p>38 1EA</p> <p>DS1302 RTC 모듈</p> <p>아두이노 등 마이크로컨트롤러에서 사용이 가능합니다.</p>	<p>39 1EA</p> <p>아두이노 우노 프로토 쉴드</p> <p>UNO 보드에서 회로를 간단히 짜기 위해 보드 위에 얹어 사용하는 쉴드입니다.</p>	<p>40 1EA</p> <p>3축 가속도 센서 모듈</p> <p>가속도를 측정할 수 있는 센서입니다.</p>
<p>29 1EA</p> <p>모터 드라이버 모듈</p> <p>ULN2003 스텝 모터 드라이버 모듈 5V ~ 12V를 사용할 수 있습니다.</p>	<p>30 1EA</p> <p>LM35 온도 센서</p> <p>온도를 마닐로그 값으로 출력합니다.</p>	<p>31 1EA</p> <p>수위 센서 모듈</p> <p>센서 액체에 잠긴 정도를 마닐로그 값으로 출력합니다.</p>	<p>32 1EA</p> <p>SG90 서보모터</p> <p>Vcc, GND, 신호선, 총 3개의 핀이 있습니다. 로봇팔이나 자동차, 비행기 조종에 사용됩니다.</p>	<p>41 1EA</p> <p>5V DC모터</p> <p>5V DC모터</p>	<p>42 1EA</p> <p>인체 감지 센서 모듈</p> <p>핵파선을 이용해 움직임 감지하는 센서입니다. 오션이 감지되면 HIGH 신호를 출력합니다.</p>	<p>43 5EA</p> <p>다이오드 1N4001</p> <p>다이오드 1N4001</p>	<p>44 5EA</p> <p>세라믹 캐패시터 (22pF)</p> <p>세라믹 캐패시터 (22pF)</p>
<p>33 1EA</p> <p>초음파 거리 센서 모듈</p> <p>5V를 사용하여 만직 거리는 2cm에서 500cm입니다.</p>	<p>34 1EA</p> <p>조이스틱 모듈</p> <p>기본적으로 조이스틱 모듈은 두개의 가변저항이 서로 수직으로 회전하는 형태로 되어 있습니다.</p>	<p>35 1EA</p> <p>온습도 센서 모듈</p> <p>아두이노 온습도 센서중 가장 대중적으로 사용되는 DHT11 디지털 센서입니다.</p>	<p>36 1EA</p> <p>RGB LED 모듈</p> <p>RGB LED 모듈로 RGB LED 세개를 하나로 묶은 상품입니다.</p>	<p>45 5EA</p> <p>세라믹 캐패시터 (1uF)</p> <p>세라믹 캐패시터 (1uF)</p>	<p>46 5EA</p> <p>트랜지스터 2N2222</p> <p>트랜지스터 2N2222</p>	<p>47 5EA</p> <p>트랜지스터 BC547</p> <p>트랜지스터 BC547</p>	<p>48 5EA</p> <p>트랜지스터 BC557</p> <p>트랜지스터 BC557</p>
<p>49 2EA</p> <p>전해 캐패시터 (50V 10uF)</p> <p>전해 캐패시터 (50V 10uF)</p>	<p>50 2EA</p> <p>전해 캐패시터 (50V 100uF)</p> <p>전해 캐패시터 (50V 100uF)</p>						

# [참고 : 저항 값 읽기]



Color	First	Second	Third	Multiplier	Tolerance
Black	0	0	0	x1	
Brown	1	1	1	x10	1%
Red	2	2	2	x100	2%
Orange	3	3	3	x1000	
Yellow	4	4	4	x10 000	
Green	5	5	5	x100 000	0,50%
Blue	6	6	6	x1 000 000	0,25%
Violette	7	7	7	x10 000 000	0,10%
Gray	8	8	8		
White	9	9	9		
Silver				x0,01	10%
Gold				x0,1	5%

